

Reducering af omkostninger og forlængelse af levetiden For Novencos produkter.

Erhvervsakademiet Dania, Skive.
Afsluttende Eksamens Opgave.

Dato for aflevering: 21. december – 2018.

Lavet af: Bent Sunesen Mortensen (bent012d)

Vejleder: Ove Thomsen.

Normalsider: 24

Opsummering

I dette projekt har jeg vurderet, hvorledes et system kan indfri Novencos ønske om at nedsætte omkostninger og forlænge levetiden for deres produkter.

I den forbindelse har jeg udviklet en prototype applikation samt database til dette formål.

Processen jeg har valgt at bruge til systemudviklingsmetode for dette projekt, er Unified Process. UP er en iterativ metode, der er opdelt i de fire faser, Inception, Elaboration, Construction og Transition.

Inception fasen har været den fase hvor meget af det redegørende og indledende arbejde er foregået. Her har jeg lavet Brief Use Cases til at få projektets omfang identificeret, samt prioriteret disse efter, hvor kritisk jeg mener implementering vil blive for hver Use Case. Dertil har jeg lavet Domænemodellen As-is for at forstå hvordan verden ser ud i dag, og så har jeg lavet domænemodellen To-be, for at give min vurdering, af hvordan fremtiden kan se ud. Ydermere har jeg lavet en virksomhedsanalyse af Novenco, for at finde frem til deres værdier.

Den tilhørende SQL-database er også blevet udviklet i Inception fasen. Her har jeg støttet mig til To-be domænemodellen, og resultatet af dette arbejde har udmøntet sig i et Entity Relation Diagram, Mapping Skema og Data Definition Language script.

Elaboration fasen, er den fase, hvor jeg har designet og kodet meget af applikationen. Jeg har inddelt Elaboration fasen i tre iterationer. Dette er gjort for at kunne arbejde med de prioriterede Use Cases i hver iteration. Jeg har i disse Elaboration Iterationer brugt forskellige metoder i udviklingen af applikationen, som f.eks. Design Studio Method, Fully Dressed Use Cases, System Sekvens Diagram og Mock Ups.

Construction fasen har været den fase hvor der er blevet arbejdet i dybden med implementeringen af designs og kode. Jeg har gennem projektet lavet manuelle funktion test, med ud over det har jeg lavet Unittest og integrationstest.

Transition fasen har bestået i at lave en alfatest og kigge overordnet på funktionaliteten af applikationen, samt at få rundet projektet af med det sidste rapport skrivning og klargøring af filer til aflevering af projektet.

Dokumenteringen af den afsluttende eksamens opgave er resulteret i denne rapport og bilag, samt den udviklede applikation og tilhørende database.

Projektperiode med start d. 22-10-2018 og slut d. 21-12-2018 samt aflevering af rapport kl. 12:00.

Indholdsfortegnelse

Indhold

Opsummering.....	1
Indholdsfortegnelse.....	2
Introduktion.....	5
Problemstilling.....	6
Problemformulering.....	6
Inception.....	7
Metode og værktøjer	7
Unified Process	7
Design Studio Method	7
Værktøjer.....	7
Integrated Development Environment	7
Andre værktøjer	8
Programmeringssprog	8
Applikationen	8
Database	8
Virksomhedsanalyse.....	9
Novenco.....	9
Domæne model.....	10
As-is model	10
To-be model.....	11
Brief Use Case.....	12
Prioritering.....	12
Database modellering	13
Entity Relation Diagram (ERD).....	13
Mapping skema	15
Database valg	16
Data Definition Language (DDL)	16
Elaboration	17
Iteration #1	17
Fully Dressed Use Cases.....	17
MockUps.....	20
MockUp Use Case #1	20

MockUp Use Case #2	21
MockUp Use Case #5	22
Sekvens Diagram	23
Iterationsresultat	24
Iterationsevaluering	24
Iteration #2	25
Use Case #4.....	26
Use Case #8.....	26
Iterationsresultat	27
Iterationsevaluering	27
Iteration #3	28
Design Studio Method	28
Fordele og ulemper	31
Fordele.....	31
Ulemper	31
Entity Relation Diagram opdatering.	32
Mapping skema opdatering.....	33
Iterationsresultat	34
Iterationsevaluering	35
Construction	35
Struktur.....	35
Mappe struktur.....	36
Database struktur	36
Vindues oversigt	37
Class Diagram	38
Interaktioner.....	39
Refaktorering.....	40
Testning	40
Manuel funktionstest	40
Integrationstest	40
Unittest	41
Construction evaluering	42
Transition.....	42
Testresultat.....	42

Spørgsmål 1.	42
Spørgsmål 2 og 3	43
Perspektivering.....	43
Konklusion	44
Bilag	46
Dagbog.....	46
Tidsplan	56
1. del.	57
2. del.	58
Entity Relation Diagram opdatering.....	59
Mapping skema opdatering.....	60

Introduktion

Denne rapport er blevet til, i et samarbejde med kaastrup|andersen. Dette er et led i k|a's plan, om at lave digitale 'Internet of Things løsninger' til deres kunder. k|a har talt med mange potentielle kunder, her i blandt Novenco, som de gerne vil lave en Proof of Concept applikation for.

Novenco er et firma som udvikler og producere car park ventilatorer til udluftning af skadelige gasarter i parkeringskældre. De arbejder på at koble deres ventilatorer på internettet, ved hjælp af Internet of Things. Novenco vil gerne kunne indhente Statusser (Condition Monitoring) for de enkelte ventilatorer.

Udgangspunktet er at en Novenco ventilator indrapporter, dens egen status, om det så er rystelser (vibrationer), strømforbrug, temperatur mm., til en server. Og derfor vil jeg i denne rapport undersøge, hvordan man kan anvende disse statusser, på en måde så det spiller ind i Novencos visioner for reducere af omkostninger og drift af ventilatorer, samt forlænge levetiden.

Jeg har lavet denne rapport, som et led i min afsluttende eksamens projekt.

Problemstilling

kaastrup|andersen (k|a) leverer forretningskritiske løsninger, der sætter deres kunder i stand til at opnå markante fordele i en teknologisk og digitaliseret verden, ved hjælp af projektleder konsulenter.

k|a kan hjælpe kunder med at lave Internet of Things (IoT). Det går essentielt ud på at få kundernes produkter koblet til internettet. k|a har i dag, en god forståelse omkring IoT og alt hvad det indebærer af protokoller, sikkerhed, elektronik mm.

k|a står over for en potentiel ny kunde, der hedder Novenco. Novenco laver ventilatorer til parkeringshuse og de har et ønske om at nedsætte udgifterne for den enkelte ventilator, da drifts- og vedligeholdelse, i produktets levetid, har omkostninger der overstiger ventilatorens indkøbspris med en faktor 20. Samtidigt kan Novencos konkurrenter masseproducerer ventilatorer, i andre lande, og lave en billigere indkøbspris, og derved mister de markedsandele.

Novenco har udtænkt den strategi at reducere omkostninger og forlænge levetiden af deres produkter, så den efterfølgende drift og vedligeholdelse vil blive reduceret. For at kunne dette, skal de have noget måleudstyr på deres produkter, som f.eks. måler temperatur vibrationer, strømforbrug, vindhastighed eller CO², ved hjælp af IoT. Det vil ligeledes skabe grundlag for at man kan lave predictive maintenance. Dette tiltag skal gerne vinde markedsandele tilbage.

k|a har ikke kendskab til Novenco infrastruktur og vil gerne illustrere, hvorledes opsamlet data, fra en ventilator, kan anvendes internt eller eksternt hos Novenco. Hvad skal der ske, når en ventilator sender en fejlmeddelelse? Og hvad sker der i processen efter en fejlmeddelelse? Hvordan kan man sikre at der er en, en til en, cause and effekt på fejlmeddelelser?

Problemformulering

Hvordan kan synliggørelsen af opsamlede data, bidrage videre i værdikæden for service tekniker, kunde eller tredjepart, således det bidrager til reducere omkostninger og forlængelse af levetiden, for Novencos produkter?

Hvordan kan jeg udvikle en applikation, som kan notificere en service tekniker om ventilations data, der overstiger fastsatte tolerance?

Hvordan kan jeg opsamle serviceteknikerens fejlrettelser fra en applikation?

Hvordan kan back end løsningen afrapportere generelle fejlrettelser til virksomheden?

Inception

Denne fase i projektet har været en analyserende fase, den har været med til at danne grundlag for det fremadrettede arbejde. Jeg har i denne fase arbejdet med Metode valg til projektet, virksomhedsanalyse, domænemodeller As-is og To-be, for at skabe en bedre forståelse af projektets domæne. Jeg har lavet Brief Use Cases for at klarlægge og definere projektets omfang.

Jeg har også gennem, arbejdet med Entity Relation Diagram og Mapping, lavet et bud på hvordan databasen kommer til at se ud. Til databasen er der også lavet DDL og mockup data, til at kunne foretage nogle visuelle test, der bekræfter mit arbejde.

Metode og værktøjer

Unified Process

Til udviklingsmetode har jeg valgt at bruge udviklingsmetoden Unified Process. Det er med til at sikre at ens arbejde er af en vis kvalitet mæssig beskaffenhed. Et aspekt af Unified Process er at man vurderer hvilke risici der kan opstå og ved at arbejde med dem tidligt i processen, kan man få afklaret disse usikkerheder, i projekt.

Agile og Scrum?

Ud fra min viden omkring projektet har jeg valgt ikke at anvende Scrum, af flere grunde. En af de grunde er måske mindre vigtig end andre, men som en-mands gruppe vil det ikke give mening at kører med en udviklingsmetode der anbefale at anvende Scrum i teams på 6 plus minus 3.

En anden grund til ikke at kører Scrum, fra et agilt synspunkt, er projektets omfang, da store projekter gerne har mange ønsker og behov og projektets omfang ikke står helt klart fra start til slut. Jeg ser derfor at have en plan drevet udviklingsmetode, som den rette metode i dette tilfælde, da omfanget af dette projekt ikke er for stort til at overskue.

Design Studio Method

Design Studio Method er en metode, man kan bruge til at generer mange konceptuelle GUI designs. Denne metode gør det muligt, via en styret brainstorm proces, at indhente designs til at løse enhver problemstilling. Metoden virker bedst, når man inddrager repræsentanter fra mange områder som f.eks. salg, udvikling, marketing, osv., ved at gøre dette får man en bred vifte af kompetencer, der alle, har forskellige indgangsvinkler, på en given problemstilling.

Værktøjer

Integrated Development Environment

Jeg vil anvende MS Visual Studio 2017, som mit IDE til udviklingen af applikationen. Grunde til dette er at det er et framework der indeholder og samler mange værktøjer i et, som f.eks. live debugging, kompilering, unittestting mm.

Jeg vil anvende Microsoft SQL Server Management Studio til at arbejde med databasen. Den giver mig gode muligheder for at oprette en database og redigere i denne.

Til hosting af databasen har jeg anvendt en Microsoft SQL server. Det har jeg gjort fordi jeg kan have en udgave af databasen til at køre på min computer, og fordi det er en relationel database, som dette projekt gør brug af.

Begge disse IDE 'er er værktøjer jeg kender godt og har arbejdet med under uddannelsen.

Andre værktøjer

Google sheets	Jeg har lavet en overordnet tidsplan for projektets forløb i google sheets. Jeg startede med at lave en vejledende tidsplan med de fire forskellige faser, som Unified Process foreskriver, Inception, Elaboration, Construction og Transition. Til koordinering af arbejdsopgave har jeg valgt at bruge min tidsplan fra google sheets, som er et spreadsheet. Se bilag – Tidsplan.
Dagbog	Jeg har valgt at føre en dagbog over projektet, da jeg så kan se tilbage på min proces. Det er på Github at jeg fører min dagbog således at den altid er tilgængelig. Github er også et godt redskab at lave dokumentation på. I dagbogen vil jeg også føre en slags dagsorden, med Næste dags gøremål og hvilket arbejde der er blevet gennemført, og arbejdet på, den aktuelle dag.
Github	Til Versionsstyring af min applikation, har jeg valgt at anvende Github. Jeg har valgt at anvende Github fordi det er online og således tilgængelig alle steder med internet, Jeg kunne også have valgt at anvende Team Foundation Server, men så skulle jeg til at sætte mig ind i hvordan dette værktøj fungerer og hvordan det skulle konfigureres. Ved at anvende Github kan jeg oprette et repository hurtigt og smertefrit under nogen særlig form for konfigurerings, samtidigt bruger Github et simpelt interface til at persistere mit arbejde, hvilket sparer mig tid, tid som jeg kan bruge på at arbejde i stedet for. Det er min intention at sikre mit arbejde, hver dag på github.
Draw.io	Til at lave diagrammer og modeller har jeg anvendt onlineværktøjet draw.io. Det er et gratis værktøj. Det indeholder en masse forskellige objekter, der gør sig gældende, inden for systemudvikling, om man så laver diagrammer, Mock Ups, SD, eller ERD, så kan man altid finde det man skal bruge.

Programmeringssprog

Applikationen

Jeg har valgt at lave applikationen som et Windows Presentation Foundation (WPF) program. Med det udgangspunkt betyder det at applikationen bliver skrevet i C#. Grunden til at jeg har valgt WPF er på grund af C#, da WPF bliver kodet i C#. C# er også et sprog jeg kender godt til og er vant til at bruge. En anden grund til at anvende C# er at kaastrup|andersen gerne vil tale ind til virksomheder i Danmark, der er et Microsoft land. På den måde er det med til at hjælpe dem i deres markedsføring af k|a.

Dertil kommer at applikationen også anvender eXtensible Application Markup Language (XAML) til at lave GUI's.

Database

Til databasen har jeg brugt Structured Query Language (SQL). SQL er det mest udbredte programmeringssprog til relationelle databaser. Det er også et sprog jeg kender godt og finde mig godt tilpas med.

Virksomhedsanalyse

Virksomhedsanalysen er lavet ud fra research på Novencos hjemmeside¹.

Novenco

Novenco er en del af Schako Group.

Essensen af Novenco

Novenco har fokus på miljøet og det aspekt har de inddraget i deres design og fremstilling af ventilationsudstyr. Ventilatorer er Novencos hovedområde og de lægger vægt på at deres produkter er ressourcevenlige, samt at Novencos engagement sikre lang levetid og en sikker miljømæssig produktionsproces. Novencos ventilatorer er en del af industrielle, kommercielle, offentlige og beboelsesbygninger over hele verden. Produkter og tjenester markedsføres og distribueres gennem Novencos omhyggeligt udvalgte datterselskaber og agenter.

Grønt fokus

Novencos forretning er koncentreret omkring de landbaserede applikationer og markeder. Det er også på dette marked Novenco udvikler og implementere deres højeffektive ventilatorer, der kræver minimale ressourcer og reducere udledningen af skadelige stoffer. Car Parks segmentet er Novencos kerneforretning, som Novenco revolutionerede med deres jet-fans i 1990'erne. Disse jet fans har ført til et stigende antal installationer. Selv i kritiske situationer i tilfælde af brand, forbliver miljøet i fokus, da mængden af energi til ventilatorerne er mindst. Disse Jet fans beskytter både liv og miljøet.

Høj kvalitet

Novenco garanterer produkter og systemer af høj kvalitet, der har høj ydeevne i mange år. Kvalitet er et nøgleord fra start til slut i deres design proces. For at bevare den høje kvalitet bliver Novencos processer kontinuerligt vurderet, for at opfylde design kravene. Samtidig tildeler Novenco den nødvendige tid og ressource til at sikre at nye designs er korrekt og verificeret.

Innovation

Siden 1947 har Novencos design udvikling og drift af ventilation produkter og systemer givet Novenco stor erfaring. Den indsats Novenco lægger i forskning og udvikling afspejler dette og gør Novenco i stand til at skabe produkter på den teknologiske front med hensyn til ydeevne og holdbarhed. I en verden, der er i konstant forandring, er det Novenco ønske at kunne levere innovative løsninger, der er miljøvenlig og lever op til den næste generation af ventilationsprodukter.

Produktionsmiljø

Novencos produkter bliver fremstillet i Danmark og er i overensstemmelse med Miljø Standarderne i henhold til ISO 9001 og ISO 14001. Novenco minimere belastningen på miljøet og reducere energiforbruget, forbedrer sorteringen af affald, minimerer stålskrot og evaluerer deres produktsortiment og leverandører fra et miljømæssigt synspunkt. Den indsats der kræves for at betjene miljøet og markederne med de rigtige produkter er konstant voksende og det er en udfordring Novenco forpligter sig til hver dag at opretholde. Alle Novencos produkter og systemer er certificeret i henhold til ISO 9001:2015 og ISO 14001:2015.

¹ <https://www.novenco-building.com/> dato: 29-10-2018

Domæne model

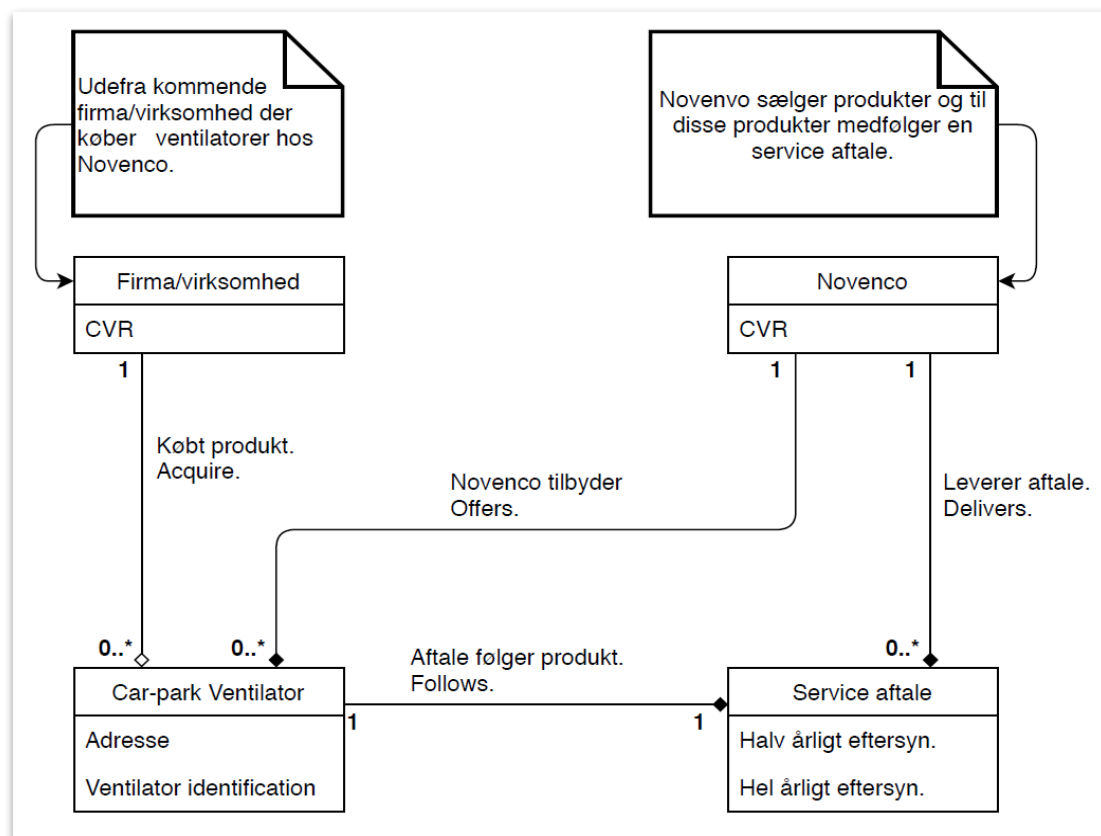
Jeg har i Inception fasen valgt at lave to domæne modeller. jeg vil lave en As-is og en To-be model. Jeg har valgt at lave de to domænemodeller for bedre at kunne forstå relationerne hos Novenco og Novencos kunder, Samt at synliggøre hvorledes de forskellige partnere interagerer med hinanden.

Udførelsen af de to modeller, er med til at skabe en dybere forståelse af hvor vi er i dag med As-is. Med To-be vil jeg prøve at vise hvad fremtiden kan tilbyde og gøre anderledes, for dermed at realisere Novencos ønsker.

As-is model

Min As-is model i Figur 1. er lavet på baggrund af de oplysninger jeg har indhentet fra kaastrup|andersen og gennem de møder k|a har haft med Novenco, samt informationer jeg har kunnet finde via Novencos hjemmeside².

Modellen repræsenterer relationen mellem Novenco og én kunde hos Novenco. Modellen viser også at der foretages en halv eller helårlig eftersyn på car-park ventilatorer, for at sikre drift og venlighold af ventilator. Denne udgift til eftersyn er en fast og kendt udgift, som er sat højt for at sikre kvaliteten af drift og vedligeholdelse.



Figur 1. Domæne model As-is.

² <https://www.novenco-building.com/> dato: 29-10-2018

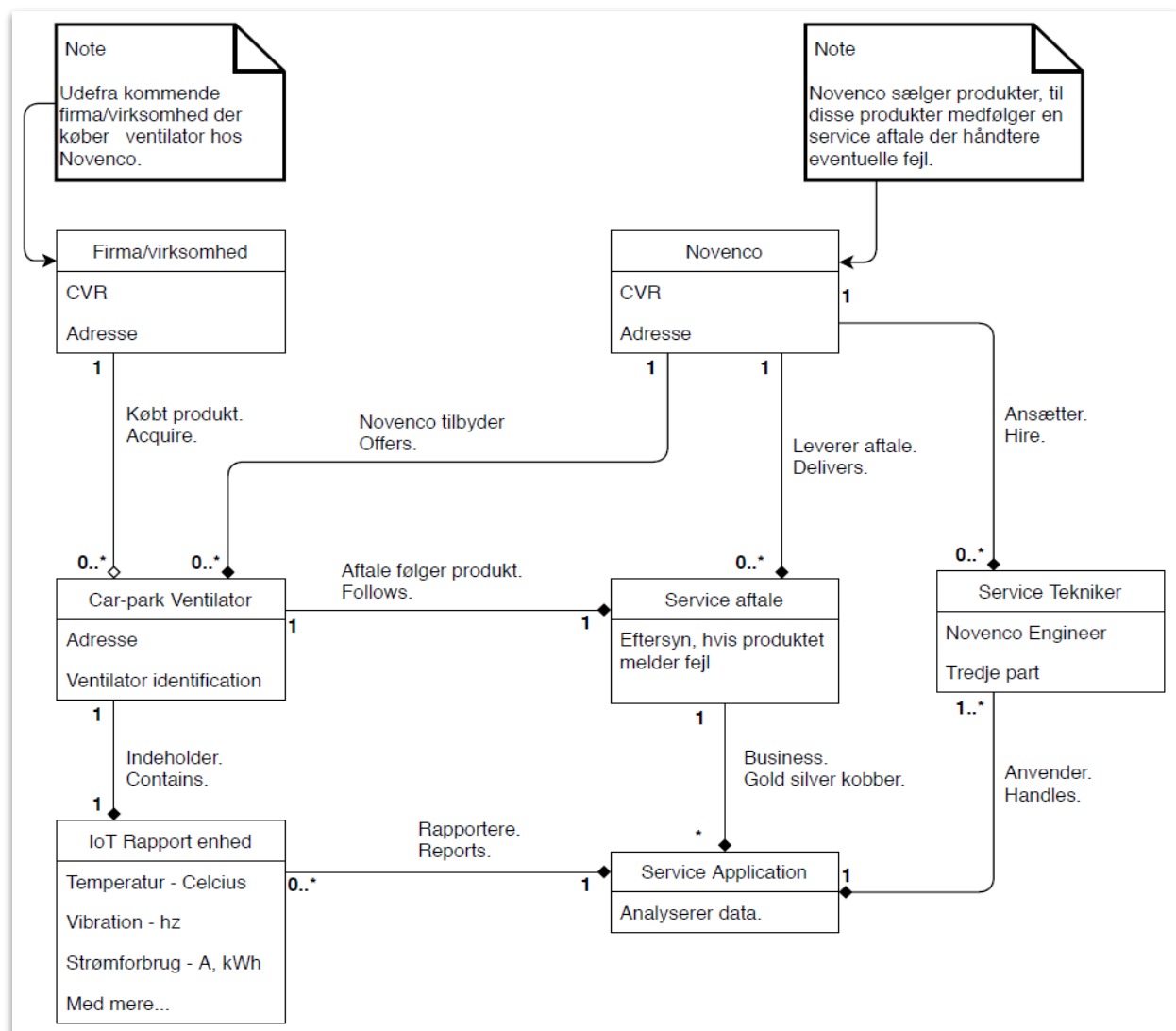
To-be model

Min To-be domæne model i Figur 2. er lavet på baggrund af de ønsker, der er fremstillet i problemstillingen. Jeg har også anvendt virksomhedsanalysen i dette skridt.

Denne To-be model er med til at vise, hvordan Novencos domæne kan komme til at se ud. Jeg har tilføjet et ekstra lag med "IoT Rapport enhed", "Service Tekniker" og "Service applikation", samtidigt har jeg ændret Service aftalen for at få en forretningsvinkel med ind over projektet.

Det ekstra lag med "IoT Rapport enhed", "Service Tekniker" og "Service applikation" er med til at kunne eliminere at skulle lave halv eller hel årligt eftersyn, på en ventilator. Hvis der kan nøjes med at lave eftersyn hvert 2. år, fordi ventilatoren først melder fejl efter 2 år, så kan omkostningerne til drift og vedligeholdelse spares væk i den periode.

Serviceaftalen har jeg ændret til at have forskellige serviceniveauer, her kan man forestille sig at en car park har en lav benyttelsesgrad og derfor gerne vil køre med en billigere service aftale, mod at denne car park vil blive nedprioriteret i forhold til en anden car park der har en høj benyttelsesgrad.



Figur 2. Domæne model To-be.

Brief Use Case

Jeg har valgt at lave Brief Use Cases Tabel 1. for at kunne danne mig et overblik af projektets omfang. Disse Brief Use Cases er lavet på baggrund af mit arbejde med domænemodellen As-is og To-be, problemstilling og problemformulering. Det er disse Brief Use Cases, som jeg vil beskrive dybere i Elaboration fasen som Fully Dressed Use Cases.

I forbindelse med det foregående afsnit kan jeg se at nogle funktionaliteter vil være til gavn for serviceteknikeren, således kan han se at systemet er aktiv. Således vil serviceteknikeren kunne se status på normalt kørende ventilatorer i en periode uden fejl, det vil skabe tillid til applikationen.

#1	Modtage fejlmeddelelser på en ventilator.	Brugeren har behov for at kunne modtage fejlmeddelelser på en ventilator, for at kunne rette fejl.
#2	Se en hurtig fejl status for en ventilator.	Brugeren har brug for at se en hurtig fejl status på en ventilator for bedre at kunne danne et overblik over hvilken fejl der er mest kritisk.
#3	Indberette fejlrettelse på en ventilator.	Brugeren skal kunne indberette, hvad årsagen til fejlen skyldes og hvilke tiltag der er blevet taget for at afhjælpe fejlen.
#4	Sætte grænseværdier for en ventilator.	Novenco skal kunne sætte grænseværdier for hvornår en ventilator skal generere en fejl.
#5	Generer mock data til demonstration.	Jeg har brug for at lave noget mock data på en ventilator til at kunne teste funktionaliteten af applikationen.
#6	Oprette yderlig fejlrettelse på en ventilator med flere fejl.	Brugeren skal have mulighed for at kunne oprette og indrapportere flere fejl på en ventilator.
#7	Oprette fejl som ikke er meldt ind af en ventilator via IoT.	Brugeren har behov for at kunne indberette ekstra fejl, som er opdaget ved synlig inspektion af en ventilator.
#8	Se normal status på en ventilator.	Brugeren skal kunne se normal status på en ventilator. For at skabe tryghed.

Tabel 1. Brief Use Cases

Prioritering

Jeg har valgt at prioritere min liste af Brief Use Cases Tabel 1., så jeg kan arbejde med en del af dem i hver iteration af Elaboration fasen. Således kan jeg fokusere på enkelte Use Cases i hver iteration af Elaboration faserne. Listen er også prioriteret efter hvilke funktion der er mest kritisk for projektet udvikling. Min prioriterede liste ser således ud Tabel 2.

Elaboration 1	Elaboration 2	Elaboration 3
#1, #2 og #5	#3, #6 og #7	#4 og #8

Tabel 2. Prioriterede Use Case liste.

Database modellering

Domænemodellen To-be Figur 2. har været en stor hjælp i mit arbejde med at lave en database til projektet. Det er ud fra den model og virksomhedsanalysen, at jeg har modelleret databasen, som den ser ud i skrivende stund.

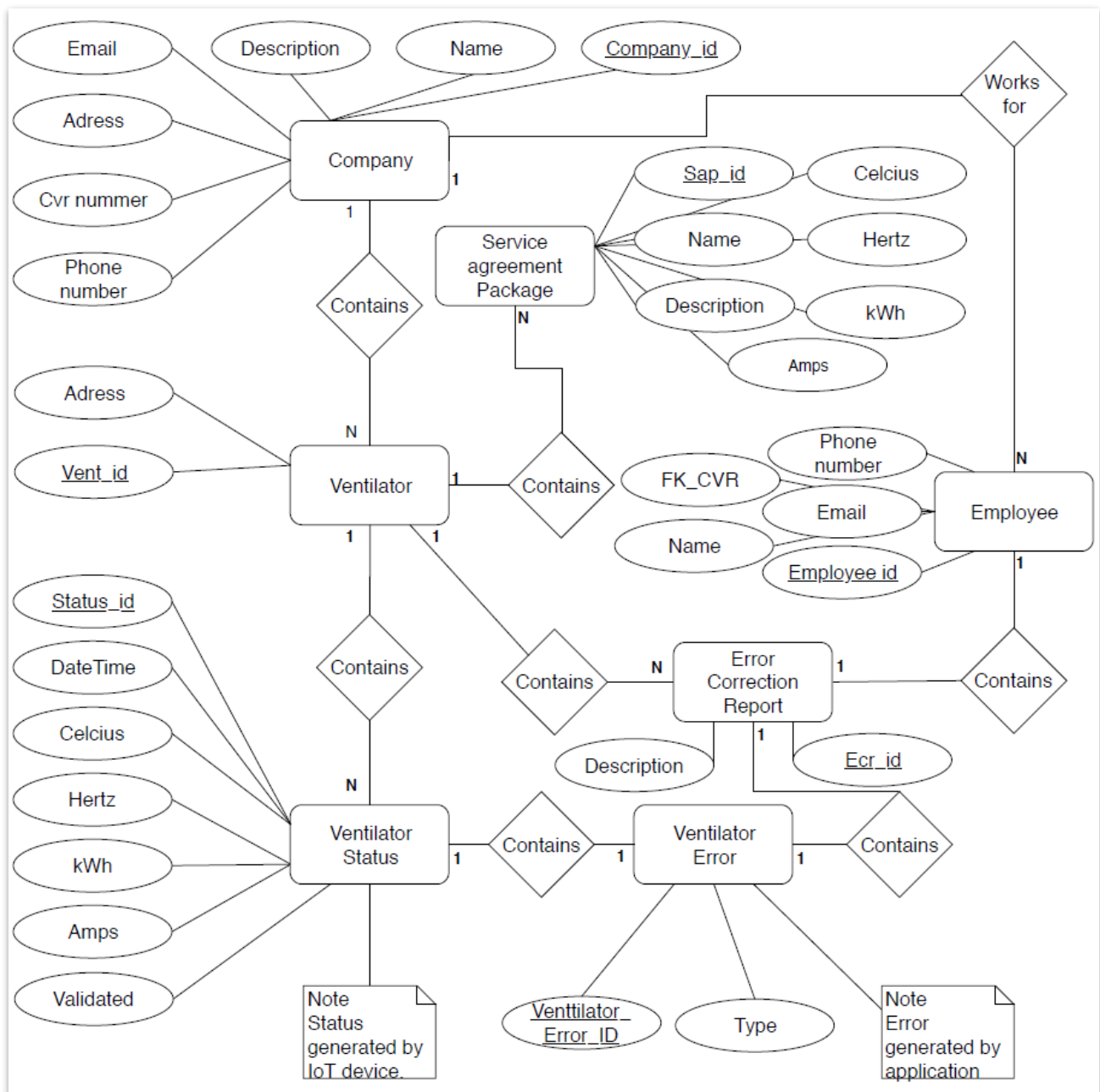
Entity Relation Diagram (ERD)

Jeg har brugt min domænemodel To-be, som min basis for denne konceptuelle database model (ERD) Figur 3. Ved hjælp af denne har jeg fundet frem til 7 entiteter, som er kernen i denne database, det har været grundlaget for min database opbygning. Essensen af denne database er at registrere fejl og fejlrettelse, i form af en rapport, indrapporteret af en Employee.

De forskellige entiteter jeg har fundet frem til, er Company, Employee, Ventilator, Service Agreement Package, Ventilator Status, Ventilator Error og Error Correction Report. Ingen af mine entiteter er svage da de alle har et unikt id, det sikre at mit system er stærkt og jeg får heller ikke lavet forhindringer for mig selv hvis jeg vil opdatere min database og tilføje nye tabeller.

Med mit ER diagram vil jeg vise disse relationer.

- Et Company har en Employee, og en Employee har et Company.
- Et Company kan have en Ventilator, og en Ventilator har et Company.
- En Ventilator kan have en Service Agreement Package, men hvis Novenco skal servicere Ventilatoren skal den selvfølgelig have en Service Agreement Package.
- En Ventilator kan have mange Ventilator Statusser, og en Ventilator Status kan kun have en Ventilator.
- En Ventilator kan have mange Error Correction Reports, og en Error Correction Report kan kun have en Ventilator.
- En Ventilator Error har en Ventilator Status, og kan have en Error Correction Report.
- En Error Correction Report har en Employee.

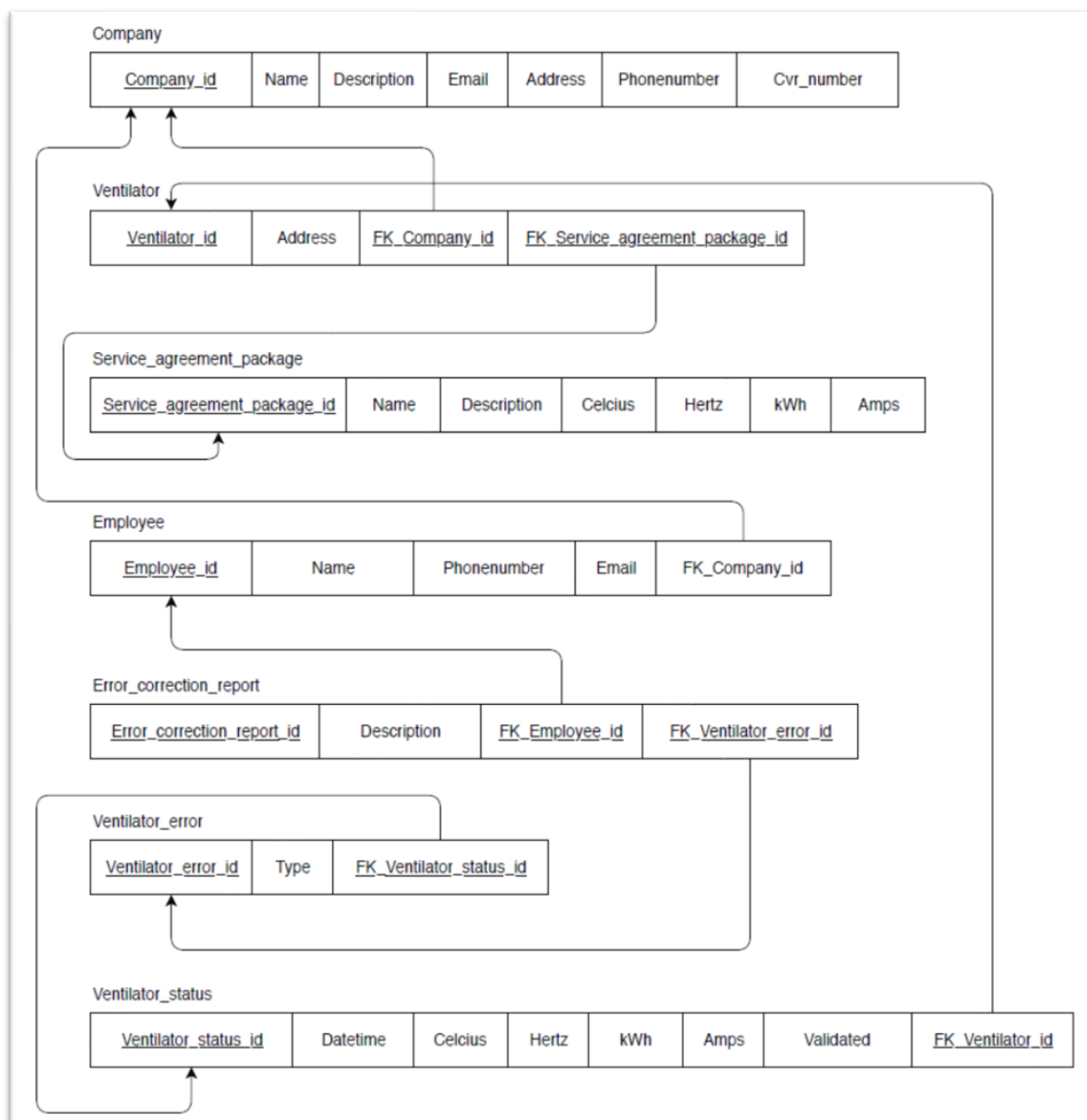


Figur 3. Entity Relation Diagram.

Mapping skema

Jeg har lavet et mapping skema Figur 4., til bedre at kunne se relationerne som de vil fremgå i en database, i modsætning til Entity relation diagrammet, hvor relationerne er markeret med streger og symboler mellem entiteterne, er det her i mapping skemaet blevet erstattet med primær-nøgler og fremmed-nøgler. Jeg har valgt at navngive primær-nøgler med tabellens navn efterfulgt af '_id' og fremmed-nøgler starter altid med 'FK_'. Jeg har valgt at gøre dette i stedet for bare at have et intetsigende navn som ID, nu er det muligt at kunne forstå databasen uden at skulle finde dokumentationen frem.

En ting som jeg har ændret fra ERD og til mapping skemaet er navngivningen af tabeller og felter. Dette er sket i forbindelse med at jeg har lavet mapping skemaet og jeg ville gerne have at det er let at genkende og adskille forskellige tabeller på tværs af databasen.



Figur 4. Mapping skema.

Database valg

Op og indtil dette skridt har processen, med ERD og Mapping, til at strukturere en database været det samme, når man skal lave en database.

Jeg har valgt at anvende en MS SQL Server til at persisterer mit data. Grunden til dette valg er at jeg har godt kendskab til platformen, det er også en relationel database, som passer ind i dette projekt. Samtidig har platformen en kæmpe bruger skare, hvilket betyder at man kan finde mange eksempler og masse af dokumentation på nettet, hvis der skulle opstå problemer med at oprette en database eller lave diverse database kald. Det er et valg der er med til at sikre min arbejdsgang, på en måde, så jeg ikke bliver forhindret i at arbejde videre på grund af problemer.

Data Definition Language (DDL)

I mit arbejde med ERD og Mapping har det været en relativ hurtig opgave at oprette en database samt lave tabeller og constraints med DDL. Med lidt online hjælp, har jeg hurtigt kunne finde løsninger på de problemer der opstod, ved at lave DDL for databasen.

Jeg har også genereret noget Mock data Figur 5 til at lægge på min database for at bekræfte at mit arbejde med ERD og Mapping også virker og at det ser ordentligt ud.

```
-- Step 4 - Creating mock data.
INSERT INTO Company ([Name], [Description], Email, Phonenumber, Cvr_number)
VALUES ('Novenco', 'Ventilation fremstilling', 'novenco@novenco.dk', '55512345', '55598765');
INSERT INTO Company ([Name], [Description], Email, Phonenumber, Cvr_number)
VALUES ('Kjeldbjerg Carpark', 'Parkeringshus', 'kbcj@kbcj.dk', '55577777', '55566666');

INSERT INTO Service_agreement_package (Sap_Name, Sap_Description, Sap_Celcius, Sap_Hertz, Sap_kWh, Sap_Amps)
VALUES ('Guld', 'Guld pakke - Høj prioritet', '60', '5', '5', '3');
INSERT INTO Service_agreement_package (Sap_Name, Sap_Description, Sap_Celcius, Sap_Hertz, Sap_kWh, Sap_Amps)
VALUES ('Sølv', 'Sølv pakke - Mellem prioritet', '80', '20', '6', '4');
INSERT INTO Service_agreement_package (Sap_Name, Sap_Description, Sap_Celcius, Sap_Hertz, Sap_kWh, Sap_Amps)
VALUES ('Kobber', 'Kobber pakke - Lav prioritet', '100', '55', '7', '5');

INSERT INTO Ventilator ([Address], FK_Company_id, FK_Service_agreement_package_id)
VALUES ('Igløvej 104, 7800 Skive', 2, 1);
INSERT INTO Ventilator ([Address], FK_Company_id, FK_Service_agreement_package_id)
VALUES ('Igløvej 104, 7800 Skive', 2, 2);
INSERT INTO Ventilator ([Address], FK_Company_id, FK_Service_agreement_package_id)
VALUES ('Igløvej 104, 7800 Skive', 2, 3);

INSERT INTO Employee ([Name], Phonenumber, Email, FK_Company_id)
VALUES ('bent mortensen', 22845214, 'bent_mortensen4@hotmail.com', 1);

INSERT INTO Error_type ([Type_name]) VALUES ('Rystelser - Hertz');
INSERT INTO Error_type ([Type_name]) VALUES ('Temperatur - Celcius');
INSERT INTO Error_type ([Type_name]) VALUES ('Ampere - Amps');
INSERT INTO Error_type ([Type_name]) VALUES ('Kilowatt-timer - kWh');
INSERT INTO Error_type ([Type_name]) VALUES ('Andet - Other');

INSERT INTO Spare_part (Spare_part_name) VALUES ('Andet');
INSERT INTO Spare_part (Spare_part_name) VALUES ('føler');
INSERT INTO Spare_part (Spare_part_name) VALUES ('lejer');
INSERT INTO Spare_part (Spare_part_name) VALUES ('motor');
INSERT INTO Spare_part (Spare_part_name) VALUES ('ravpluks');
INSERT INTO Spare_part (Spare_part_name) VALUES ('varmeskjold');
```

Figur 5. Mock data.

Elaboration

Elaboration fasen er en del af Unified Process, det er her at en stor del af analyse arbejdet til selve applikationen kommer til at foregå. Mange af de ting der er blevet arbejdet og fundet frem til i Inception fasen, vil nu blive uddybet og bearbejdet i denne fase.

Elaboration fasen er, for dette projekt, blevet delt op i tre iterationer. jeg vil analysere de mest kritiske funktionaliteter i første iteration, ved at få bearbejdet disse funktionaliteter tidligt i processen kan jeg bedre få et billede af om projektet kan blive gennemført med et tilfredsstillende resultat. Nogle Use Cases kan også være afhængig af, at andre Use Cases er blevet lavet færdige, så derfor skal disse også laves tidligt i processen.

Mit mål med hver iteration er at dokumentere og implementere de prioriterede Brief Use Cases. For at opnå dette har jeg i næste afsnit beskrevet, hvorledes jeg vil håndtere det og hvilke værktøjer jeg vil benytte.

Iteration #1

I modsætning til Inception fasen, som gik meget på at skulle forstå og finde frem til funktionaliteter af programmet, så vil der i Elaboration fasen også være en del analyse arbejdet.

Jeg vil lave Fully Dressed Use Cases af de prioriterede Brief Use Cases, til denne første Elaboration iteration. Jeg vil også lave System Sekvens Diagrammer for en enkelt Fully Dressed Use Case for at kunne danne mig et billede af hvordan interaktionen mellem program og database ser ud.

I denne fase vil jeg også komme med bud på hvordan programmet kommer til at se ud, ved hjælp af Mock Ups. Disse Mock Ups bliver lavet af de forskellige prioriterede Brief Use Cases det vil gøre arbejdet lettere, når jeg skal til og lavet GUI i WPF.

Fully Dressed Use Cases.

Jeg har lavet disse Fully Dressed Use Cases for bedre at kunne danne mig et overblik over hvordan applikationen kommer til at se ud rent design mæssig. Jeg har kort noteret i min dagbog, hvilke tanker eller udfordringer der er på dette stadie, det er lidt ligesom hønen og ægget, hvad kommer først, men jeg tror ikke at det gælder om at komme først med noget her, men mere at få det til at spille sammen i en større sammenhæng. Med det sagt betyder det at der har været mange ændringer i Fully Dressed Use Cases og Mock Ups.

Jeg har under denne proces samtidigt tænkt meget over hvordan at programmet bliver bygget op design mæssig og hvordan det kommer til at give den bedste, bruger oplevelse, i anvendelsen af applikationen.

Af de prioriterede Brief Use Cases har jeg lavet 3 Fully Dressed Use Cases i denne iteration

- # 1 - Modtage fejlmeddelelser på en ventilator. Tabel 3.
- # 2 - Se en hurtig fejl status for en ventilator. Tabel 4.
- # 5 - Generere mockup data til demonstration. Tabel 5.

Tabel 3. Fully Dressed Use Case #1.

Use Case name	#1 - Modtage fejlmeddelelser på en ventilator
Scope	Novenco
Level	Bruger
Primary Actor	Bruger
Stakeholders & Interests	Bruger som gerne vil se en liste over ventilatorer med fejl
Pre-conditions	Brugeren har adgang til internettet.
Success guarantee	Brugeren får vist en liste med ventilatorer hvor en fejl er registreret
Main success scenario	<ol style="list-style-type: none"> 1. Brugeren åbner appen. 2. Brugeren vælger sit eget "navn", i en dropdown "Service tekniker". 3. Brugeren bliver vist en liste over ventilatorer med fejl.
Extension	2a. Brugers navn findes ikke. <ol style="list-style-type: none"> 1. Brugeren anmoder om oprettelse. 2. Brugeren bliver oprettet af systemadministrator. 3. Brugeren vælger sit eget "navn", i en dropdown "Service tekniker". 4. Brugeren bliver vist en liste over ventilatorer med fejl.
Special requirements	-
Technology and data variation list	-
Frequency of occurrence	Lav
Miscellaneous	-

Tabel 4. Fully Dressed Use Case #2.

Use Case name	#2 - Se en hurtig fejl status for en ventilator.
Scope	Novenco
Level	Bruger
Primary Actor	Bruger
Stakeholders & Interests	Bruger som gerne vil se en hurtig fejl status, til vurdering af hvor kritisk fejlen er.
Pre-conditions	Brugeren har adgang til internettet.
Success guarantee	Brugeren får vist en fejl status for den valgt ventilator
Main success scenario	<ol style="list-style-type: none"> 1. Brugeren trykker på "Se fejl status". 2. Systemet åbner et nyt lille vindue, med relevante informationer om fejlen. 3. Brugeren kan åbne yderligere fejl statusser og sammenligne fejl statusser.
Extension	
Special requirements	-
Technology and data variation list	-
Frequency of occurrence	Lav
Miscellaneous	-

Tabel 5. Fully Dressed Use Case #5.

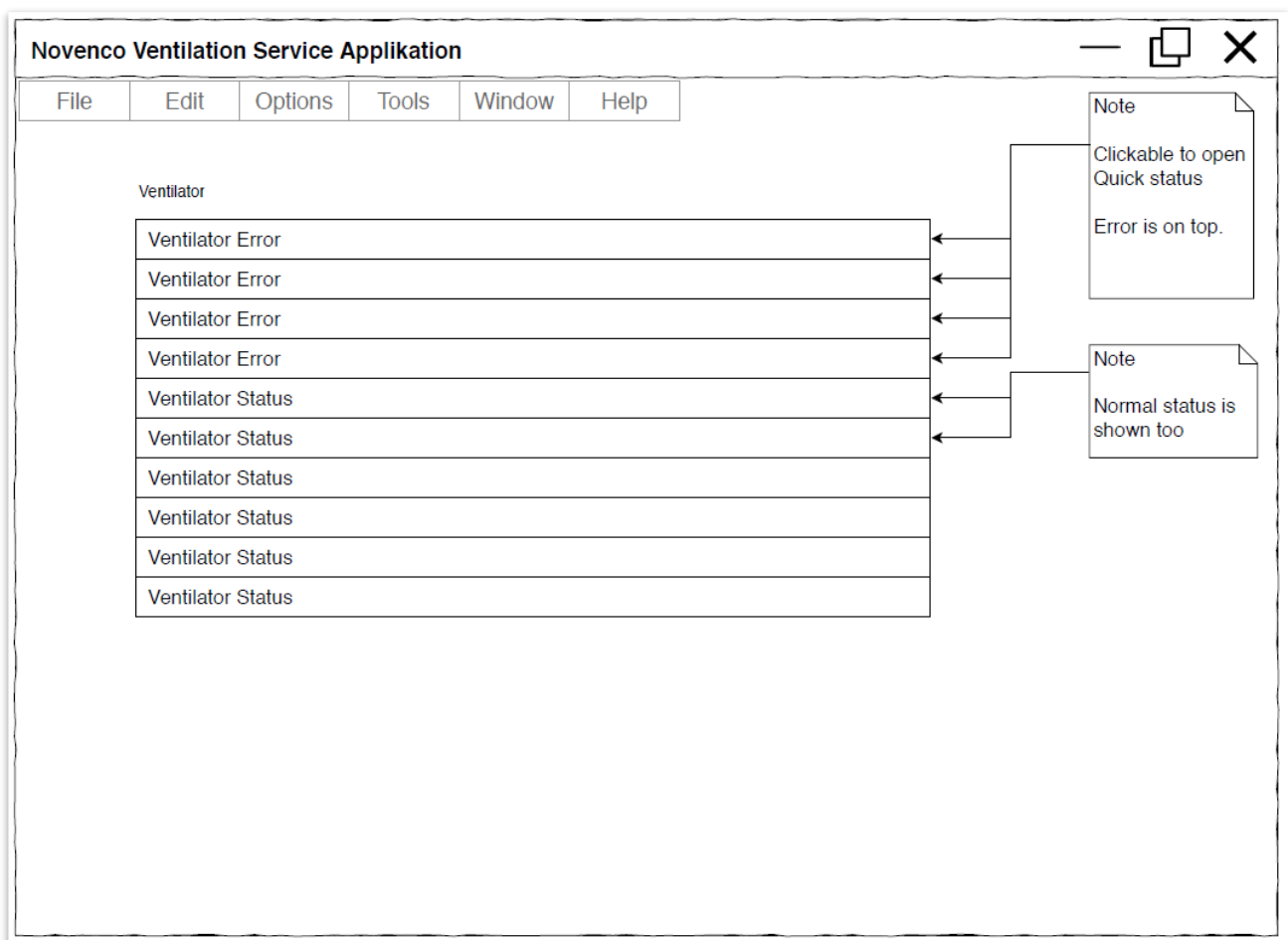
Use Case name	#5 – Generere mock data til demonstration.
Scope	Novenco
Level	System og Bruger
Primary Actor	Bruger
Stakeholders & Interests	System
Pre-conditions	En fuldt fungerende database. Internet.
Success guarantee	Status for en ventilator bliver persisteret i en database.
Main success scenario	<ol style="list-style-type: none">1. Brugeren afvikler program.2. Brugeren trykker på "Open mock status generator"3. Brugeren trykker på "Generate status"4. Systemet sender nu med en tilfældigt genererede status til databasen.
Extension	
Special requirements	-
Technology and data variation list	-
Frequency of occurrence	Lav
Miscellaneous	-

MockUps

Til at lave MockUps har jeg anvendt onlineværktøjet Draw.io. Jeg har til inspiration af designet, brugt Fully Dressed Use Cases til applikationen. Det sikre at der er overensstemmelse med Fully Dressed Use Cases og MockUps. Samtidigt gør det også at der er nogle klare retningslinjer for hvad applikationen skal udtrykke og vise med de forskellige GUI interfaces.

MockUp Use Case #1

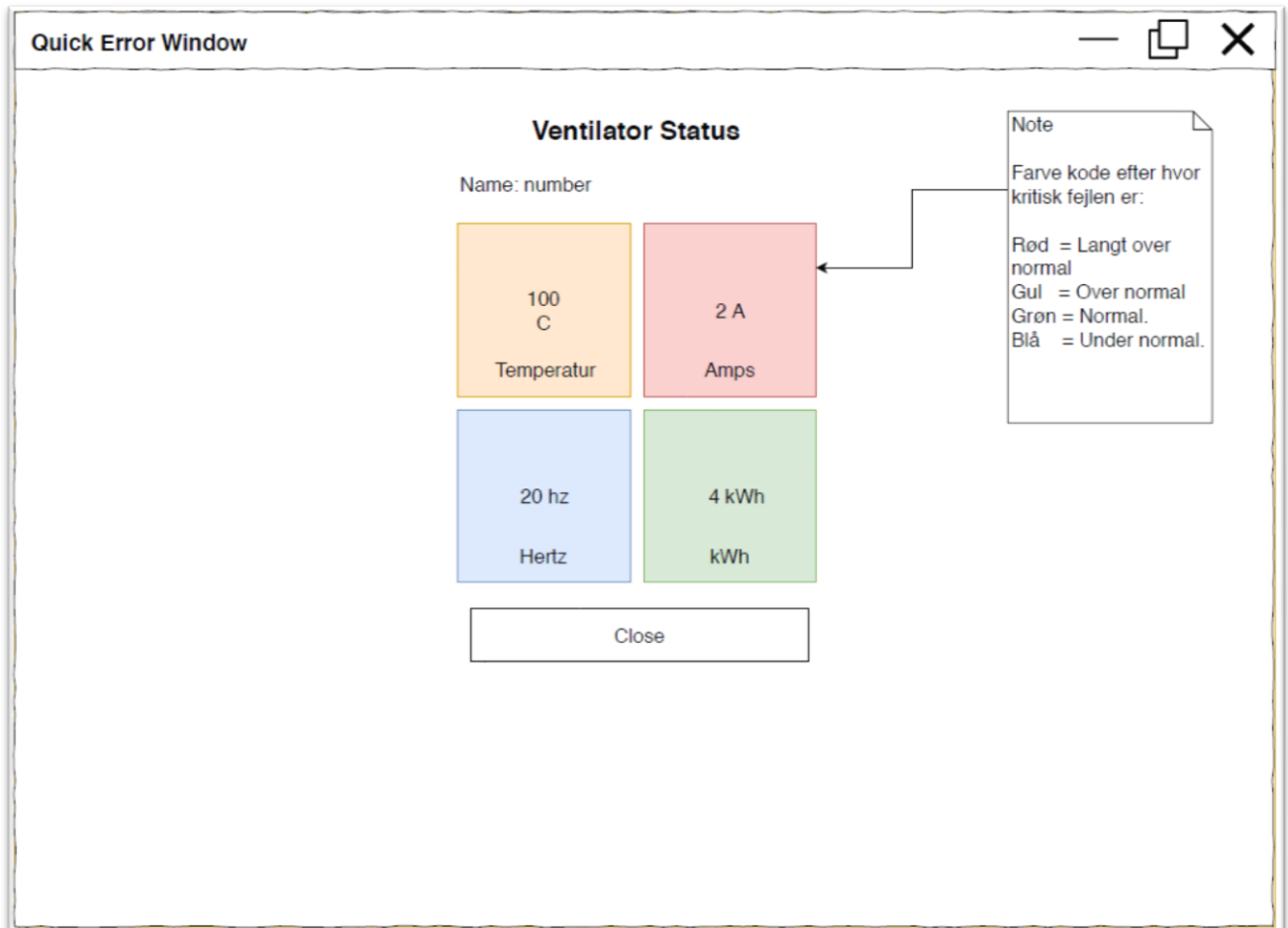
Til Use Case #1 har jeg lavet denne MockUp Figur 6. Grund ideer har været, at når man starter applikationen op, bliver man præsenteret med dette GUI interface som fremstiller en liste med Ventilatorer. Ventilatorerne bliver listet op, således at dem med en registreret fejl vil figurere øverst på listen med ventilatorer. Hele delen med at modtage fejlmeddelelser, foregår ved at åbne denne applikation. Der vil så ligge en funktion i applikationen som lytter på databasen efter statusser, som kan ligge udenfor den fastsatte grænse.



Figur 6. MockUps. StartSide.

MockUp Use Case #2

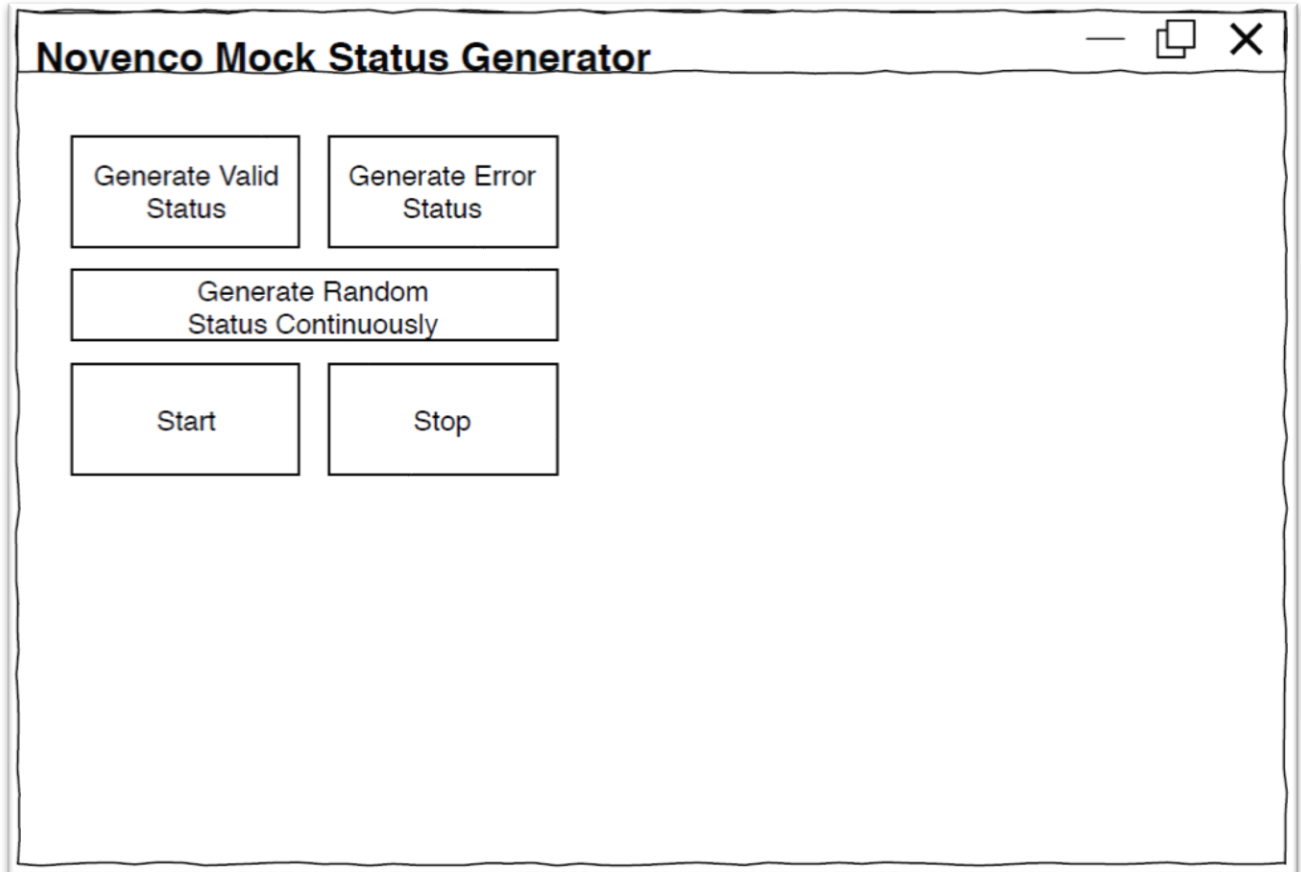
Til Use Case #2 har jeg lavet denne MockUp Figur 7. Den viser en simpel, men nem at forstå, ventilator status for en ventilator. Service teknikeren kan således åbne flere statusser og holde dem op imod hinanden, her kan han så planlægge sin dag ud fra lokationer mellem ventilatorer eller efter hvilken ventilator der kræver eller har brug for mest opmærksomhed.



Figur 7. MockUp Use Case #2.

MockUp Use Case #5

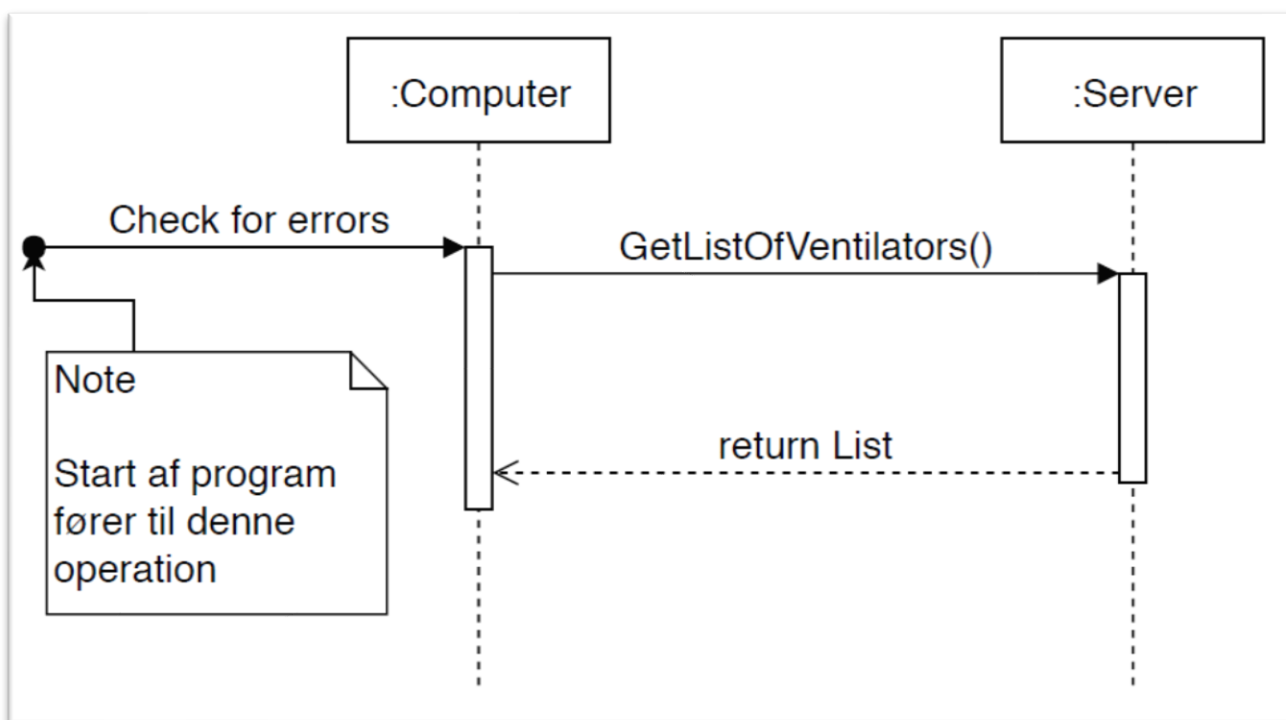
Til Use Case #5 Figur 8., har jeg lavet en simpel applikation der skal simulere datastrømmen fra en ventilator med IoT. Den kommer til at bestå af nogle simple knapper, der kort beskriver hvilken data, ventilator status, der bliver persisteret på databasen. Den er lavet for at kunne generere statusser og teste applikationen.



Figur 8. MockUp Use Case #5.

Sekvens Diagram

Jeg har lavet et sekvensdiagram Figur 9., for at godt i gang med udviklingen af applikationen. Sekvensdiagrammet viser interaktioner arrangeret i tidssekvenser, den viser de objekter og klasser, der er en del af scenariet. Man får en forståelse for rækkefølgen af meddelelser udvekslet mellem objekter, der er nødvendige for at udføre scenariets funktionalitet.



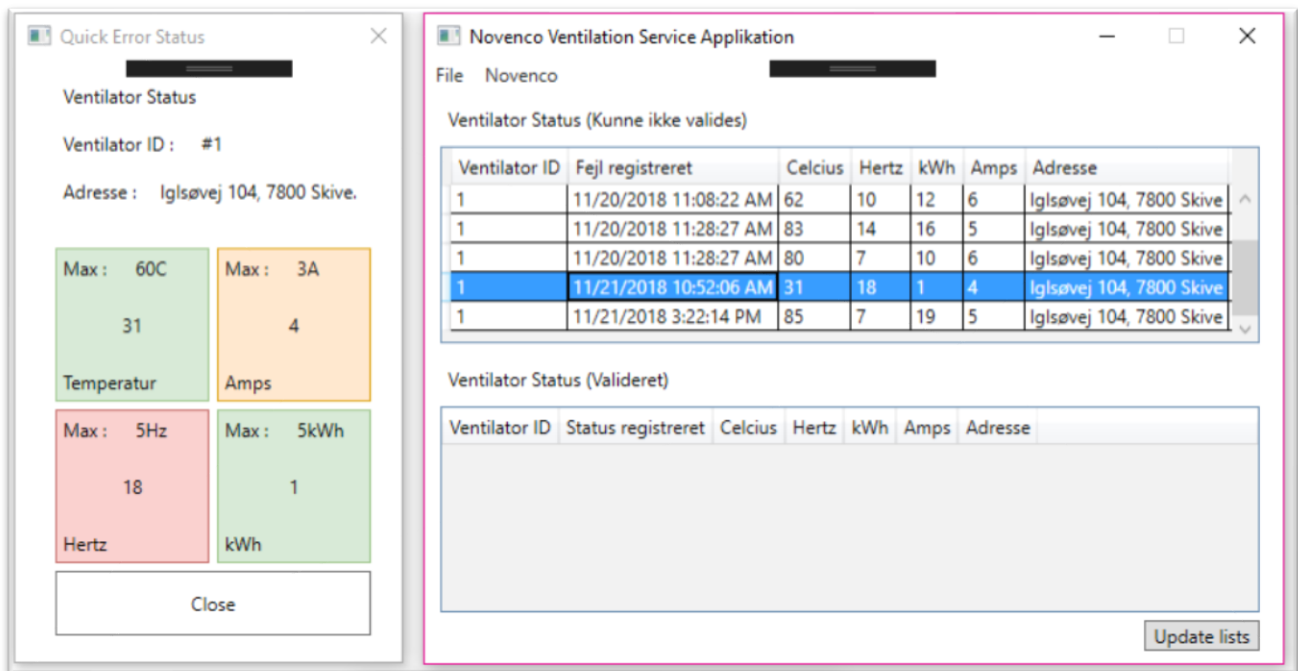
Figur 9. Sekvensdiagram Use Case #1.

Alle udregningerne for om en status, skal generere en fejl eller ikke, foregår på brugerens computer. Her er det vigtigt at applikationen henter data fra serveren omkring grænseværdier og sammenligner disse op mod de målte værdier der kommer fra en ventilator. En ventilator har en service agreement package, der indeholder grænseværdier.

Iterationsresultat

Jeg vil nu vise resultatet af iteration #1. og beskrive det nærmere.

Applikationen Figur 10. er blevet designet ud fra Mock Ups og Fully Dressed Use Cases. Som det fremgår af Figur 10. er der sket små ændringer i forhold til de genererede Mock Ups. Quick Error Status vinduet har også fået nogle ekstra informationer koblet på for at give mere information til brugeren. Novenco Ventilation Service Applikation vinduet er blevet delt i to lister, hvor en indeholder statusser med registreret fejl og den anden indeholder ventilatorer uden fejl, som derefter forsvinder igen når man opdatere listerne.



Figur 10. Applikation resultat iteration #1.

Iterationsevaluering

I denne første Iteration har jeg lavet Fully dressed Use Cases, Mock Ups af Use Cases til applikationen og et enkelt sekvensdiagram. Det har været en svær opgave at lave det hele selv og det har da også taget meget længere tid end jeg havde regnet med, en ting jeg synes der er gået godt er rapport skrivning og at huske at føre dagbog over projektet.

Jeg har overskredet min tidsplan med 1 uge og 2 dage. Jeg har brugt for meget tid på at kode applikationen op og har derfor været presset på tiden, her i denne Iteration #1.

I næste Iteration vil jeg ikke lave Fully dressed Use Cases, Mock Ups eller Sekvensdiagram for at få mere tid til applikationen. Dette har jeg besluttet for at prøve at indhente det tabte. Jeg vil fortsætte med at føre dagbog over projektet.

I næste iteration #2 vil jeg se hvad jeg kan gøre for at rette op på situationen.

Iteration #2

Tiden er desværre gledet fra mig og det er ikke lykkedes mig at følge min vejledende tidsplan for projektet. Grundende til dette har været en fejl vurdering af tidsestimater for projektet, når man arbejder alene. Ydermere er projektet også blevet forsinket fordi jeg ikke har haft en klar plan for hvordan interfacet skulle se ud og bygges op rent grafisk, forstået på den måde, der har ikke været nogen planer for, hvordan man navigere rundt i applikationen, og det har derfor også spillet ind i, hvorfor projektet er forsinket, på nuværende tidspunkt.

I skrivende stund er jeg midt i uge 47, og som det fremgår af Figur 11. Udsnit af tidsplan., skulle jeg gerne være færdig med Iteration #2 og fuld i gang med iteration #3, her i Elaboration fasen.

Tidsplan			
	45	46	47
	elaboration	elaboration	elaboration
	Skriv kode	Skriv kode	Skriv kode
	Fully dressed usecase	Fully dressed usecase	Fully dressed usecase
	Mockups	Mockups	Mockups
	Rapport	Rapport	Rapport
	Dagbog	Dagbog	Dagbog
21. D.			

Figur 11. Udsnit af tidsplan.

Som nævnt i iterationsevalueringen, fra forrige iteration, vil jeg nu se, hvad jeg kan gøre for at indhente min tidsplan igen. jeg har i næste iteration #3. besluttet at bruge værktøjet Design Studio Method i samarbejde med kaastrup|andersen. Dette skal hjælpe mig med det grafiske aspekt og resultatet skulle ende ud i mange konceptuelle Mock Ups til Use Cases #3, #6 og #7. og formentlig give mig nok feedback til at komme godt fra start.

På grund af møde og andre aktiviteter, i k|a, bliver denne Design Studio Method afholdt torsdag kl. 9:00 i Uge 47, hvor alle indkaldte deltagere har tid. Men det betyder at jeg skal have noget andet på tidsplanen.

På grund af denne forsinkelse og tiltag med Design Studio Method, har jeg valgt at lave om i min prioritering af Use Cases. Jeg kan måske indhente noget af den spildte tid. I iteration #3 skulle jeg arbejde med #4 og #8, Disse to Use Cases er prioriteret mindst i forhold til sværhedsgraden, og jeg vurderer derfor at disse kan laves på mindre tid.

Alt dette betyder at der skal laves nogle rettelser til min prioriterede liste af Use Cases.

Den originale prioriterede Use Case liste ser således ud Tabel 6.

Elaboration 1	Elaboration 2	Elaboration 3
#1, #2 og #5	#3, #6 og #7	#4 og #8

Tabel 6. Original prioriterede Use Case liste.

Den nye prioriterede Use Case liste kommer derfor til at se således ud Tabel 7.

Elaboration 1	Elaboration 2	Elaboration 3
#1, #2 og #5	#4 og #8	#3, #6 og #7

Tabel 7. Revideret prioriteret Use Case liste.

Dette betyder dog at jeg har valgt at lægge nogle arbejdsopgaver, der potentielt har en større risiko for at noget kan gå galt senere i processen, hvilket ikke er optimalt.

Use Case #4

Use Case #4 Tabel 8. er rettet mod Novenco og er lavet for at kunne lave en business case til forretningen. Ved at kunne sælge ventilatorer med forskellige Service Agreement Pakker eller rettere sagt service aftaler, kan de lave en forretning ud af dette. De kan samtidig lave rettelser for hvornår eller hvor tidligt en ventilator skal melde fejl, ud fra de grænseværdier, der er blevet sat. Jeg har valgt at have tre grundpakker, som udgangspunkt. Disse pakker er Guld Sølv og Kobber, Guld pakken har lavere grænseværdier for, hvornår en ventilator status skal registreres som en fejl, og Sølv er mellem trinnet og kobber er den med de højeste grænseværdier.

#4	Sætte grænseværdier for en ventilator.	Novenco skal kunne sætte grænseværdier for hvornår en ventilator skal generere en fejl.
----	--	---

Tabel 8. Brief Use Case #4.

Use Case #8

Til Use Case #8 Tabel 9. har jeg hurtig opdaget at jeg kunne genbruge noget af det jeg allerede har programmeret op til Use Case #2. Det har derfor været en hurtig opgave at lave.

#8	Se normal status på en ventilator.	Brugeren skal kunne se normal status på en ventilator. For at skabe tryghed.
----	------------------------------------	--

Tabel 9. Brief Use Case #8.

Iterationsresultat

Jeg vil nu vise resultatet af iteration #2. og beskrive det nærmere.

På billedet til venstre Figur 12. kan man angive nye grænseværdier for en service pakke. Applikationen viser de gamle værdier, og når man trykker på "Sæt nye værdier" bliver de gamle værdier i databasen opdateret.

På billedet til højre kan service teknikeren se normal status for en ventilator, her er det gjort klart med farven grøn at denne ventilator status er i orden.

Sæt nye grænseværdier

Service agreement package - Guld

Guld pakke - Høj prioritet

Gamle værdier	Nye værdier
Celcius 60 C	<input type="text"/>
Hertz 5 Hz	<input type="text"/>
kWh 5 kWh	<input type="text"/>
Amps 3 A	<input type="text"/>

Sæt nye værdier

Quick Error Status

Ventilator Status

Ventilator ID : #1

Adresse : Igløvej 104, 7800 Skive.

Max : 60C 51 Temperatur	Max : 3A 1 Amps
Max : 5Hz 0 Hertz	Max : 5kWh 0 kWh

Close

Figur 12. Applikation resultat iteration #2.

Iterationsevaluering

Denne iteration er gået helt efter hensigten. Jeg har fået implementeret 2 Use Cases, uden større besværligheder. Det har til dels været på grund af den lave sværhedsgrad af de 2 Use Cases og at der har været mulighed for at kunne genbruge dele af applikationen. Jeg har også fået lagt en plan for at komme bedre i gang med næste iteration. Ved at anvende værktøjet Design Studio Method, kan jeg måske få en bedre start på processen i næste iteration.

Iteration #3

Jeg vil i denne iteration anvende værktøjet Design Studio Method til at generere konceptuelle Mock Ups til Use Casesene #3, #6 og #7. Jeg har indkaldt interessenter Figur 13, inklusiv mig selv, til møde, i k|a, for at hjælpe med denne opgave.

Ved at have byttet om på Iteration #2 og #3, har jeg opdaget at det også er vigtigt at persistere grænseværdierne. Med applikationens nuværende funktioner, kan grænseværdier ændre sig, derfor vil historiske data ikke give mening, hvis disse grænseværdier ændre sig. Jeg har valgt at lægge disse værdier på "Error_correction_report" for at gemme værdierne på det tidspunkt hvor de er aktuelle for en fejl.

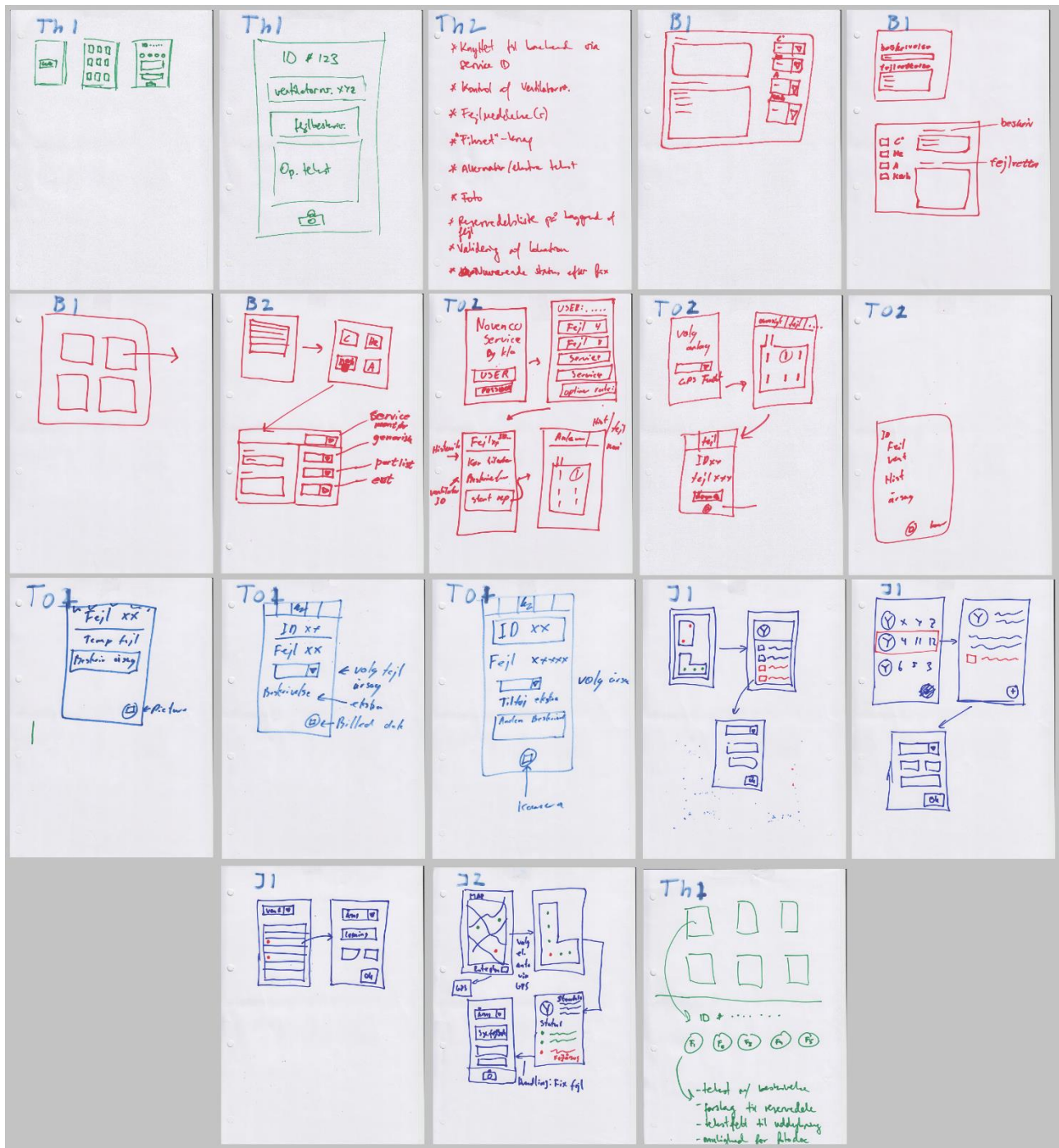
Design Studio Method

Jeg har valgt at bruge Design Studio Method værktøjet, fordi det har været en tidskrævende opgave at skulle designe applikationen selv. Ved at have denne brainstorm øvelse, som Design Studio Method er, har jeg, i samarbejde med interessenter fra k|a Figur 13, på under en time fået genereret 17 forskellige Mock Ups til applikationen Figur 14.

kaastrup andersen – IoT Team		
 Jannik Andersen Director in charge	 Thomas Kinnari IoT Architect	 Torben Godesken IoT Specialist
<p>Jannik Andersen hjælper virksomheder med kompleksitetsreduktion og procesoptimeringer, der muliggør øget effektivitet og merværdi til interne og eksterne kunder.</p> <p>Jannik har erfaring med IT, IoT og digital transformation. Jannik anvender sin funktionelle ekspertise til at udfordre etableret tænkning, og gennem top down analyser understøtter design og realisering af nye tjenester, nye funktionalitet og nye teknologiske løsninger for at sikre optimeret tid til marked og tidlig ROI.</p> <p>I kaastrup andersen er Jannik ansvarlig for den forretningsmæssige drift inden for IT, IoT og Digitalisering og er en del af ledelsen.</p>	<p>Thomas Kinnari bidrager med dyb viden om fysiske og digitale systemer ud fra hans baggrund inden for fysik, SW udvikling og smarte produkter</p> <p>Thomas har arbejdet med:</p> <ul style="list-style-type: none">• Digital service og app udvikling for finansielle kunder• Teknisk ledelse af produktudvikling inden for smarte målesystemer• Systemarkitektur for asset tracking løsning• Digitaliseringsprojekt for kunde i personsikkerhedsbranchen• Ledelse af diverse software og hardware projekter• Etablering af teknologi-roadmap inden for IoT og Digitalisering for stor teknologikunde	<p>Torben Godesken bidrager med hands-on tekniske indsigt kombineret med en stærk faglig og teoretisk ballast</p> <p>Torben arbejder med:</p> <ul style="list-style-type: none">• Visualiseringer og konceptudvikling i Seebo• Dyb forståelse for kommunikationsprotokoller• Hardware og Software udvikling• Elektriske systemer og udvikling af elektronik• Research og analyser og tekniske systemer• Test og testplanlægning• Indkøring og skalering af tekniske løsninger

Figur 13. Interessenter fra k|a.

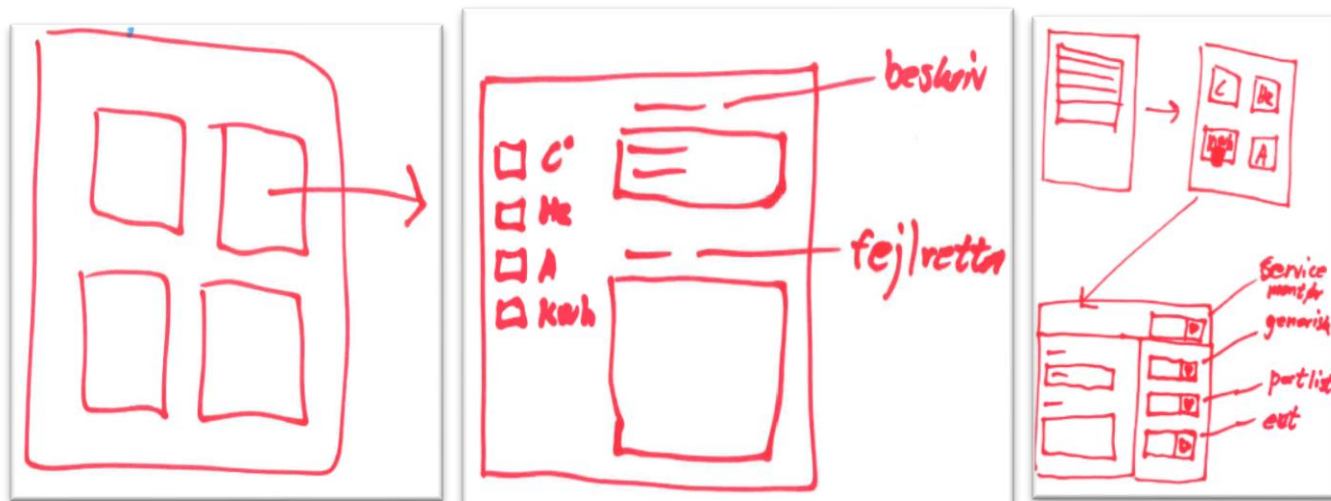
Dette er billederne, som er blevet genereret under Design Studio Method.



Figur 14. Design Studio Method resultat.

Under DSM-sessionen fandt vi frem til yderligere funktionaliteter/krav til Use Case #3, f.eks. som at en bruger skal kunne vælge reservedele og at en fejl status skal have en fejlmeddelelse. Disse funktionaliteter kan hjælpe til med at standardisere fejlrettelsesrapporter.

Jeg har her valgt at vise et udsnit af det samlede resultat³, af Design Studio Method sessionen. Figur 15. viser udsnittet med de designs, der komplimenterer applikationen bedst og løser de nye funktionaliteter. Disse designs vil lægge til grund for applikationens udseende fremadrettet.



Figur 15. Design Studio Method designs.

³ Bilag – Design Studio Method resultat

Fordele og ulemper

Jeg vil her komme ind på nogle fordele og ulemper ved DSM, og hvad min oplevelse af at anvende dette værktøj og hvad jeg har oplevet ved at lave denne øvelse.

Fordele

En fordel ved at anvende DSM er tidsaspektet, på én time, er der blevet lavet 18 forskellige designs til applikationen, samtidigt er noget af tiden gået med at forklarer processen. Så hvis jeg havde mere tid, end den time der blev afsat til DSM, havde udbyttet også været større.

Vidensdeling er også en fordel, fordi vi var 4 personer, med vidt forskellige baggrunde, kom der også mange forskellige bud, bud som jeg ikke selv havde overvejet at lave.

En anden fordel er at jeg senere kan vise en applikation hvor deltagerne kan genkende dele af deres egen designs i det færdige resultat.

Dialogen, under DSM er helt klart en fordel, det foregår i en positiv tone og alle forslag er velkomne.

Ulemper

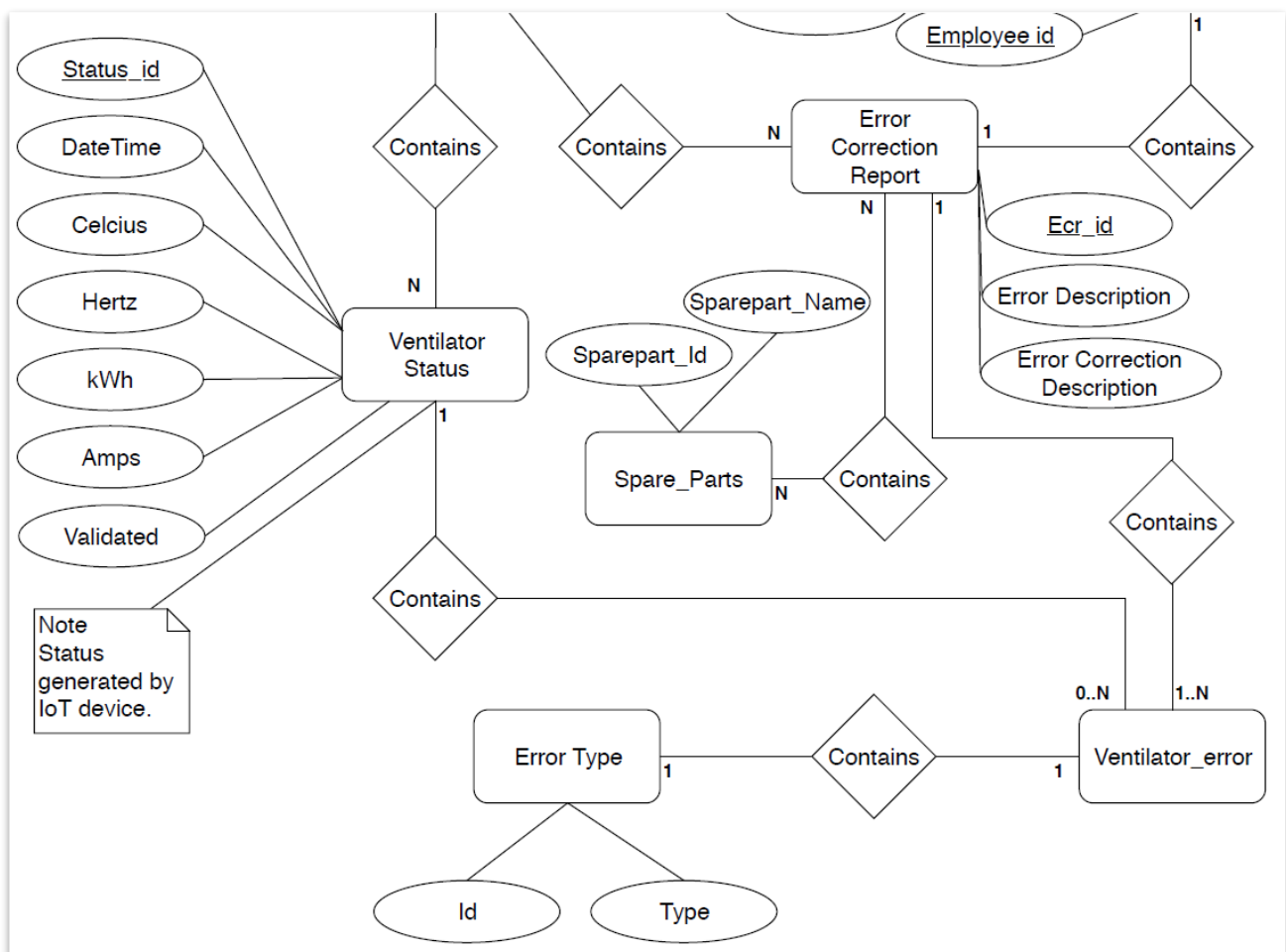
En ulempe, som jeg oplevede ved at lave DSM var at omfangsområdet, blev misforstået, eller også er det ikke lykkedes mig at formidle omfangsområdet i sådan en grad, så deltagerne designer et bestemt vindue (WPF-page). Selvom jeg læste Use Casen op og senere hen viste mit forløb lige arbejde med applikationen, blev designs med applikationens struktur, navigation fra vindue til vindue, ved med at dukke op. Hvis området havde handlet omkring, hvorledes en bruger navigere rundt i applikationen, havde nogen af de designs der blev produceret under DSM, været aktuelle.

Manglende træning. Det var helt klart også en faktor at holdet ikke havde prøvet at lavet DSM før, og det viste sig også, i og med, at en person begyndte at skrive tekst i stedet for at lave designs.

Entity Relation Diagram opdatering.

Der er på baggrund af Design Studio Method, blevet fundet nye funktionaliteter. For at imødekomme de nye funktionaliteter, har det været nødvendigt at lave ændringer til databasen, da den gamle database ikke imødekom de nye funktionaliteter, derfor vil jeg opdatere ERD. Jeg vil blandt andet tilføje reservedels lister og en bedre mulighed for at klassificere og generalisere fejl efter type.

Her på Figur 16.⁴ har jeg taget et udsnit af de ændringer jeg har lavet til ERD for bedre at kunne forstå, hvilke ting jeg skal implementere og hvordan det kommer til at ændre strukturen i databasen. Fuld ERD kan findes i bilag.



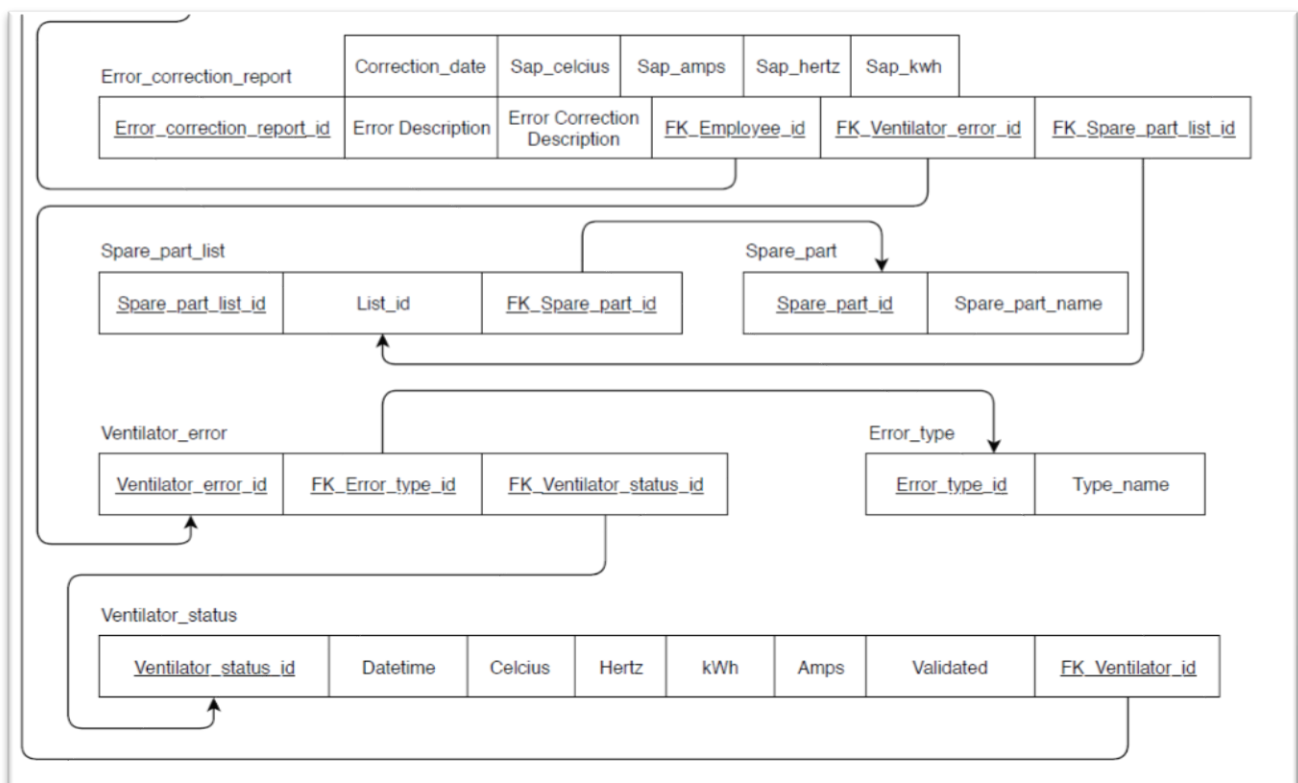
Figur 16. Entity Relation Diagram udsnit.

⁴ Bilag - Entity Relation Diagram.

Mapping skema opdatering

Jeg bliver på baggrund af de ændringer jeg har lavet til ERD, også nød til at opdatere mit Mapping skema. Jeg har også her taget et udsnit af mit Mapping skema Figur 17⁵.

Under arbejdet for denne gik det op for mig at jeg skal have persisteret de gældende grænseværdier for en fejlrettelse. Disse grænseværdier har jeg valgt at lægge på den eksisterende tabel "Error_correction_report". Grunden til dette valg, er at jeg kun er interesseret i grænseværdierne på det tidspunkt en fejl bliver rettet. Dertil har jeg også valgt at have en property "Correction_date", på "Error_correction_report", der indeholder et datostempel for, hvornår en fejl bliver rettet. Det betyder at man nu kan udlede, hvor lang tid der er gået fra fejlen, er blevet registeret og til at den er blevet rettet. Fuld Mapping Skema kan findes i bilag.



Figur 17. Mapping skema genbesøg.

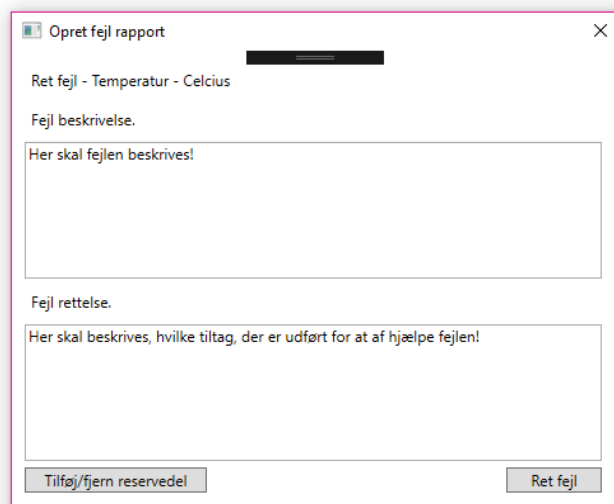
⁵ Bilag - Mapping Skema.

Iterationsresultat

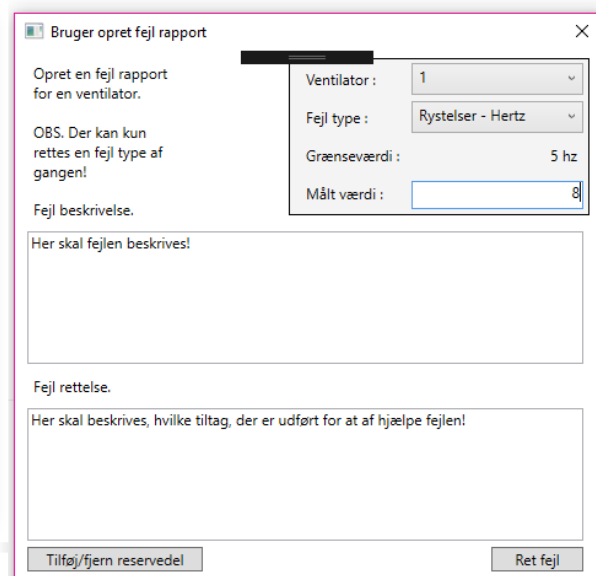
Jeg vil nu vise resultatet af iteration #3. og beskrive det nærmere.

Figur 19., viser et vindue hvor der specifikt står udspecificeret, hvilken fejl man er ved at lave en fejl rapport for. Det er et simpelt vindue hvor man skal beskrive fejlen, og komme med bemærkninger omkring fejlen og hvordan den er opstået. Man skal også beskrive hvilke tiltag der er blevet lavet for at rette fejlen. Dertil kan man også tilføje reservedele til en fejl rapport og til sidst kan man trykke på "Ret fejl", som så gemmer alle relevante detaljer som dagsdato for rettelsen, reservedelsliste, beskrivelser, gældende service agreement pakke værdier og fejl type i databasen samt hvilken servicetekniker der har oprettet fejl rapporten.

Figur 18., har samme funktion, som Figur 17., forskellen her er at på dette vindue bliver man nødt til selv at vælge ventilator id samt hvilken fejl type man vil rette. Det er blevet lavet for at kunne generere en ventilator status, som hører til en fejl rapport.



Figur 19. Applikation resultat Iteration #3 Use Case #3 og #6



Figur 18. Applikation resultat Iteration #3 Use Case 7.

Iterationsevaluering

Jeg har i denne iteration fået færdiggjort de resterende Use Cases. De 3 Use Cases jeg har prioriteret at lave i denne iteration, har alle noget at gøre med at kunne indrapportere en fejlrettelsesrapport.

#3	Indberette fejlrettelse på en ventilator.	Brugeren skal kunne indberette, hvad årsagen til fejlen skyldes og hvilke tiltag der er blevet taget for at afhjælpe fejlen.
#6	Oprette yderlig fejlrettelse på en ventilator med flere fejl.	Brugeren skal have mulighed for at kunne oprette og indrapportere flere fejl på en ventilator.
#7	Oprette fejl som ikke er meldt ind af en ventilator via IoT.	Brugeren har behov for at kunne indberette ekstra fejl, som er opdaget ved synlig inspektion af en ventilator.

Ved at indrapportere en fejl ad gangen, bliver Use Case #3 og Use Case #6 opfyldt. Use Case #7 bliver også opfyldt, ved at man kan lave en fejlrettelsesrapport uden at den pågældende ventilator selv har autogenereret en ventilator status.

Jeg har i denne iteration anvendt Design Studio Method og jeg synes at det har været en stor hjælp, Jeg har kunnet fokusere mere på at kode applikationen, i stedet for at sidde og designe applikationen op fra bunden.

Construction

Unified Process er en iterativ udviklingsmetode, som leverer inkrementelle stykker virkende kode, ved brug af disciplinerne "Business Modeling, Requirements, Analysis & Design, Implementation, Test og Deployment".

Construction fasen er den fase hvor største delen af disciplinen "Implementation" (kode arbejde) foregår, men med det i mente så er disciplinen "Implementation" også foregået i Elaboration fasen. Jeg vil derfor i dette afsnit komme nærmere ind på implementeringen og konstruktionen af applikationen, samt hvorledes jeg håndtere test af min applikation.

Det er også mit mål at alle Use Cases er blevet fuldt implementeret og færdiggjort i en tilfredsstillende tilstand i denne fase.

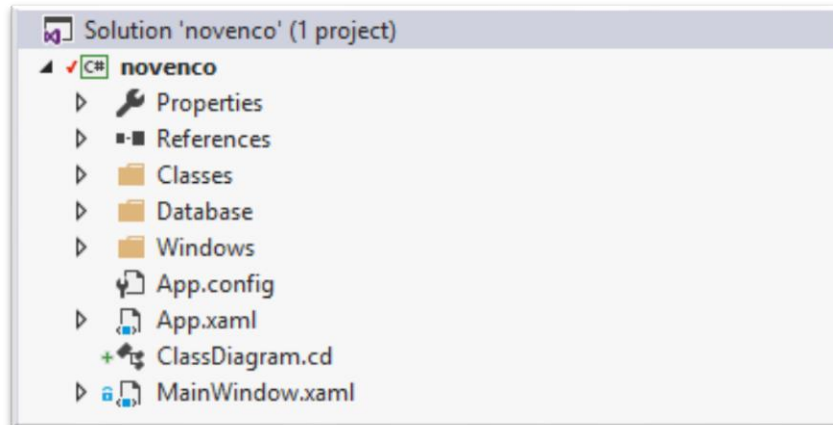
Jeg har valgt kun at have en iteration i Construction fasen

Struktur

Jeg vil i dette afsnit komme ind på nogen af de valg jeg har taget i konstruktionen af applikationen og hvordan jeg har valgt at bygge min struktur op i applikationen, som f.eks. Class diagram fra Visual Studio, vindues oversigt, mappe struktur, Interaktioner og database struktur.

Mappe struktur

Her på Figur 20. vil jeg vise hvordan jeg har opdelt mine Classes og WPF-pages (Windows) og Database i forskellige mapper. Jeg har valgt denne fremgangsmåde, for at gøre det overskueligt og ligetil at finde et givent element i applikationen.



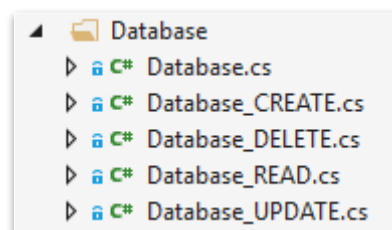
Figur 20. Mappe struktur.

Database struktur

På Figur 21. er der vist, hvordan jeg har valgt at håndtere Data Manipulation Language (DML). Jeg har valgt at inddele databasen i Database_Create, Database_Read, Database_Update og Database_Delete (CRUD). Jeg har valgt at gøre dette på grund af projektets størrelse og fordi det er begrænset hvor mange databasekald jeg skal have i min applikation.

Databaseklassen er en static partial class, hvor "Database.cs" kun indeholder attributten connection og metoderne OpenConnection(), CloseConnection() og CreateParam(). De øvrige Metoder i databaseklassen er inddelt i de før nævnte databaseclasser. Jeg har valgt at navngive de forskellige databasekald, ud fra den funktion de udfører på databasen, f.eks. som en INSERT statement vil blive navngivet med "Store" (gem) efterfulgt af "ErrorCorrectionReport" alt efter hvad denne metode skal persistere i databasen.

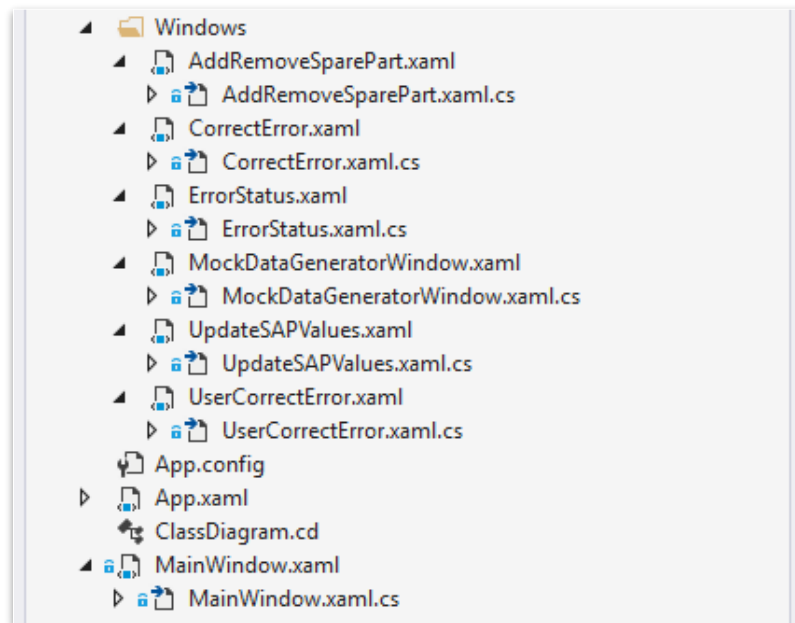
Jeg har også valgt at gøre brug af SQL-parameter. For at sikre databasen mod SQL-Injections.



Figur 21. Database struktur.

Vindues oversigt

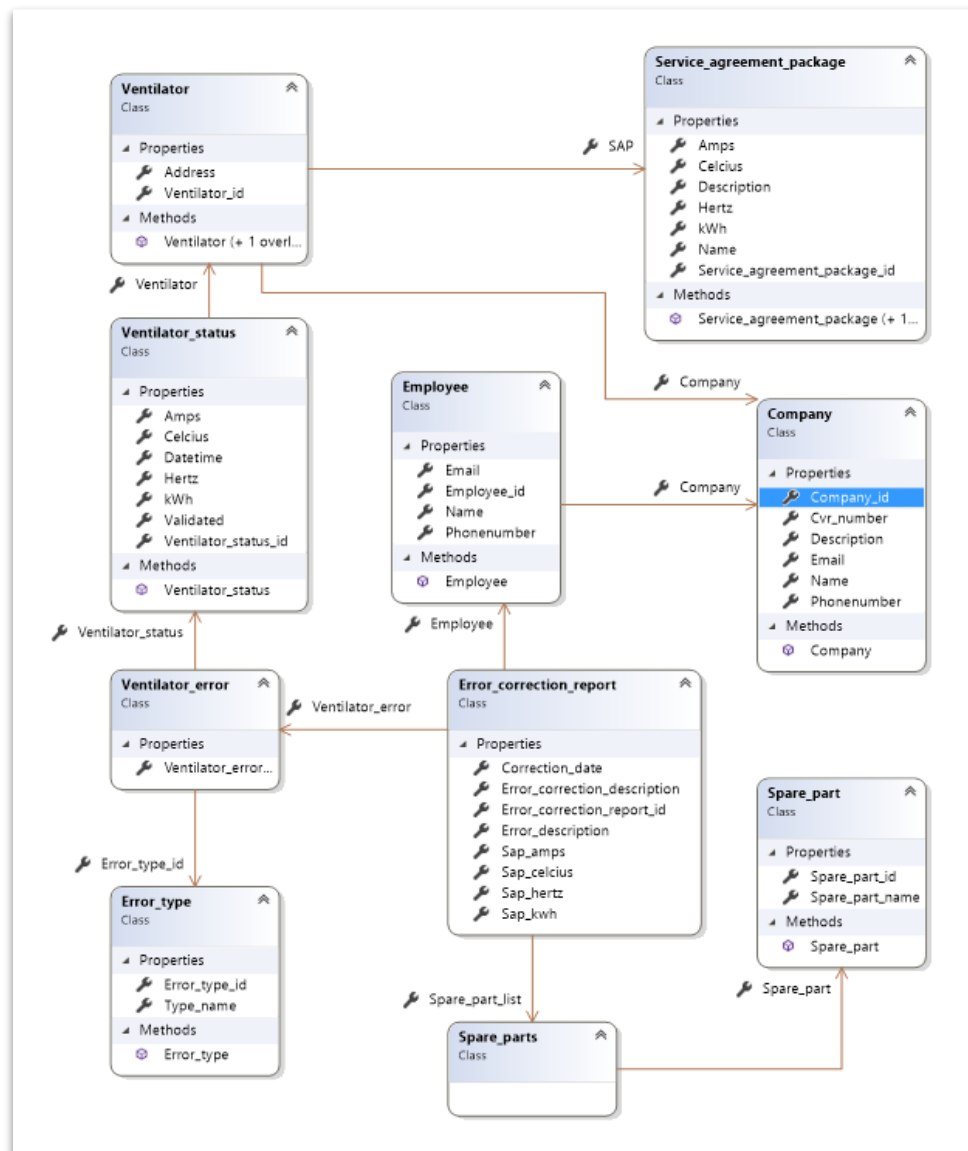
Figur 22. viser alle mine vinduer (WPF-pages). Logikken til hvert vindue ligger henholdsvis individuelt i `*.xaml.cs` filerne også kaldet "CodeBehind". Jeg har valgt at holde logikken til hvert vindue i dens egen Classe, for at skabe et bedre overblik og for at holde det simpelt. Det har også den effekt at jeg ikke deler metoder mellem vinduer og Classer.



Figur 22. Vindues oversigt.

Class Diagram

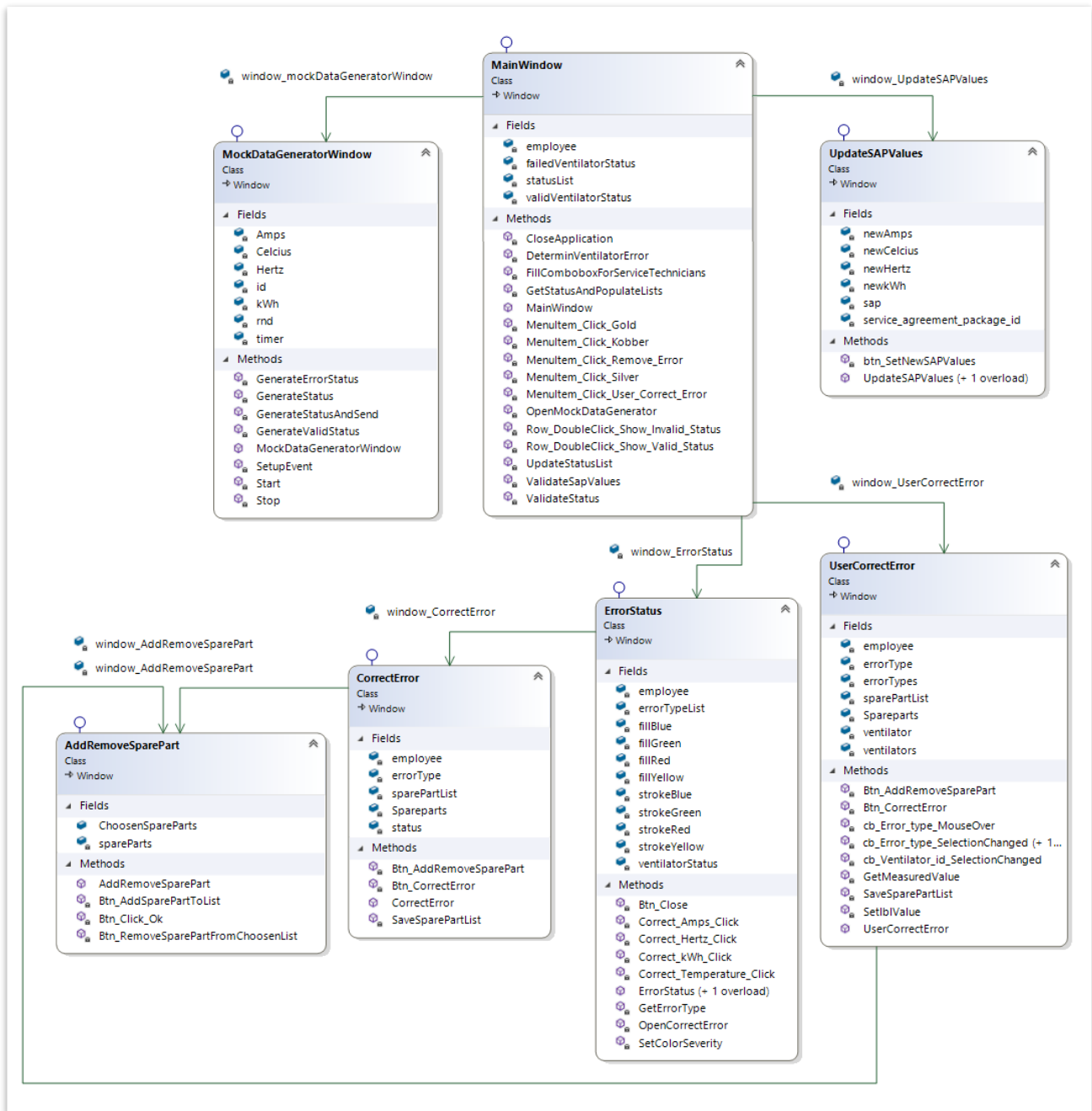
Figur 23. viser et billede af mine Classer og hvordan de hænger sammen i applikationen. Dette Class Diagram, fra Visual Studio, stemmer rigtig godt overens med min konceptuelle model fra To-be og ERD. Den afspejler at det arbejde jeg har lavet, ikke er stukket af og blevet en anden løsning end den jeg har haft som udgangspunkt i starten af projektet. Den er også god at have som dokumentation, hvis en anden udvikler skal lave tilføjelser til applikationen.



Figur 23. Class Diagram.

Interaktioner

Jeg vil her, på Figur 24., vise et Tree-diagram over interaktioner mellem vinduerne i applikationen. På figuren kan man også se hvilke attributter og metoder, hvert vindue gør brug af, i applikationen. Applikationen starter i MainWindow og grener sig derfra ud i de forskellige vinduer.



Figur 24. Class diagram - Window Interaktioner.

Refaktorering

Her i dette afsnit vil jeg beskrive hvorfor jeg har lavet refaktorering af min applikationskode. I begyndelsen af applikationens livscyklus har jeg lavet metoder, med mange linjer kode, som håndtere mange properties og bearbejder meget data på en gang. Hvilken har gjort det svært at finde rundt i koden og at lave ændringer.

Ved at lave refaktorering af applikationskoden kan jeg øge kvaliteten på en række punkter, som f.eks.

- Kan jeg øge forståelsen af min kode markant, og som udvikler betyder det at behovet for teknisk dokumentation bliver minimeret for at forstå koden.
 - Og med øget forståelse, falder fejlraten på ny funktionalitet.
- Kan jeg få mulighed for at indfører unittest af koden. Ved at metoder bliver mindre og mere testbar.
- Bedre mulighed for at finde og rette fejl.
- Bedre forståelse af koden for nye udviklere.

Testning

Manuel funktionstest

I løbet af disciplinen "Implementation" (kode arbejde), har jeg løbende testet metoder af ved hjælp af manuelle funktion tests, Det vil sige jeg først har skrevet én metode og så afviklet denne metode, for visuelt at kunne verificere at jeg får det forventede resultat. Ulempen ved denne fremgangsmåde, er at jeg hurtigt kan komme til at ændre på en metode og utilsigtet kan ødelægge en anden metoder der afhænger af den metoder der blev ændret.

Integrationstest

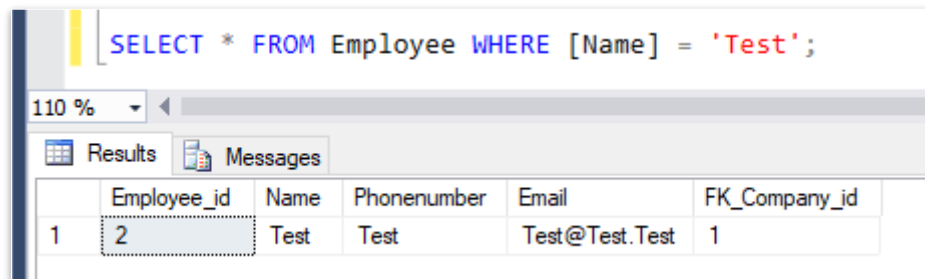
Jeg har helt i starten af projektet lavet en simpel integrationstest. Denne test er blevet lavet for at sikre at jeg har en forbindelse mellem min applikation og databasen, testen på Figur 25. Persistere en "Employee" med værdierne "Test", til databasen.

```
// Til test af forbindelse til databasen
0 references | Bent Sunesen Mortensen, 18 hours ago | 1 author, 1 change
public static void StoreTestConnection()
{
    int success = 0;
    SqlCommand command = new SqlCommand("INSERT INTO Employee([Name], Phonenumber, Email, FK_Company_id)" +
        " VALUES('Test', 'Test', 'Test@Test.Test', 1)", connection);

    try
    {
        OpenConnection();
        success = command.ExecuteNonQuery();
        CloseConnection();
        MessageBox.Show("Connection was made " + success.ToString(),
            "Warning", MessageBoxButtons.OK, MessageBoxIcon.Question);
    }
    catch (Exception)
    {
        MessageBox.Show("TestConnection failed", "Warning", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

Figur 25. Integrationstest.

Efter at overstående metode succesfuld er blevet afviklet, i min applikation, har Jeg anvendt MS-SQL Server Management Studio, Figur 26. til at verificere at min "Employee" med "Test" værdierne er blevet persisteret.



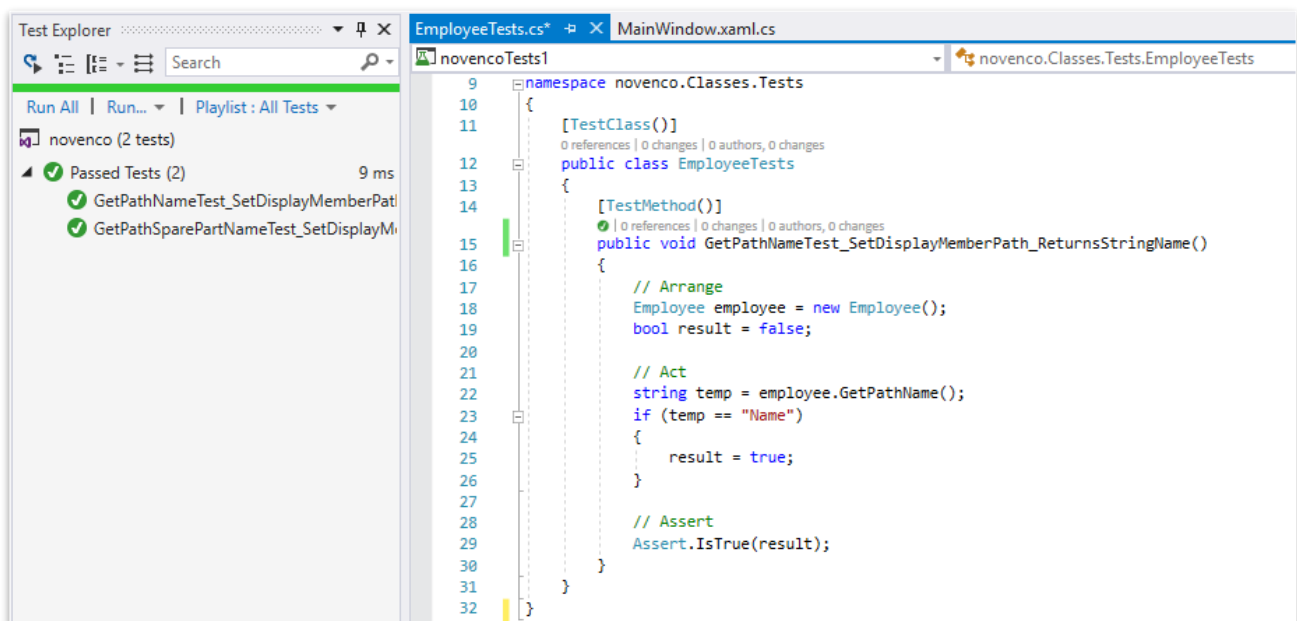
The screenshot shows a SQL query window with the text: `SELECT * FROM Employee WHERE [Name] = 'Test';`. Below the query, the 'Results' tab is active, displaying a single row of data. The columns are Employee_id, Name, Phonenumber, Email, and FK_Company_id. The row contains the values 1, 2, Test, Test, and Test@Test.Test respectively.

	Employee_id	Name	Phonenumber	Email	FK_Company_id
1	2	Test	Test	Test@Test.Test	1

Figur 26. Integrationstest - verificering.

Unittest

Jeg har lavet et par unittest på henholdsvis "Employee" og Spare_part Classen. Unittestene verificerer at jeg får fat i en string til DisplayMemberPath for en combobox attribut i min XAML-kode. Unittest er med til at nedbringe tiden i at lave manuelle funktions test af koden og kan ved større projekter minimere fejl og tidsforbrug. Figur 27. viser min test af metoden GetPathName().



Figur 27. Unittest af Employee metode.

Construction evaluering

Jeg har i denne ene Construction iteration haft som mål, at alle mine Use Cases skal være færdige, og det mål er lykkedes. Jeg har derfor en færdig prototype, som er klar til overlevering.

Jeg har gennem iterationen blandt andet arbejdet på at holde en ren struktur gennem applikationen og fået denne struktur dokumenteret her i rapporten.

Hertil kan jeg tilføje at jeg har lavet refaktorering af applikationen, for at højne kvaliteten kodemæssigt. Refaktorering har også ført til at jeg bedre kan lave Unittest fordi metoderne er blevet mindre. Ud over refaktorering og unittest har jeg også lavet ændret i navngivning, for properties og metoder, så koden er lettere at læse.

Transition

I min Transition iteration, skal applikationen og tilhørende database overleveres til kunden, men før den bliver overleveret, vil jeg lave en alfatest af min applikation. Alfatesten er lavet ud fra et brugerperspektiv og den skal se på brugervenligheden og selve applikationens funktionalitet.

Jeg har valgt at lave denne alfatest for at klarlægge, en række punkter som brugervenlighed, og applikationens funktionalitet.

Testresultat

Spørgsmål som er kommet frem under alfatesten.

1. Hvorfor forsvinder en "ventilator_status" ikke, når alle fejl er rettet, for en ventilator status?
2. Hvordan kommer man ind og retter en fejl for en "ventilator_status"?
3. Hvad med at lave noget hjælp af en art?

Med udgangspunkt i disse spørgsmål har jeg kigget på hvad jeg kan lave af forbedringer til applikationen, for at løse disse spørgsmål.

Spørgsmål 1. er en ny Use Case Tabel 10., og den bør nok ligge under perspektivering i stedet for, da det er en funktionalitet der ligger uden for dette projekts omfang. Jeg har taget den med her for at pointere at det er en prototype applikation jeg har udviklet.

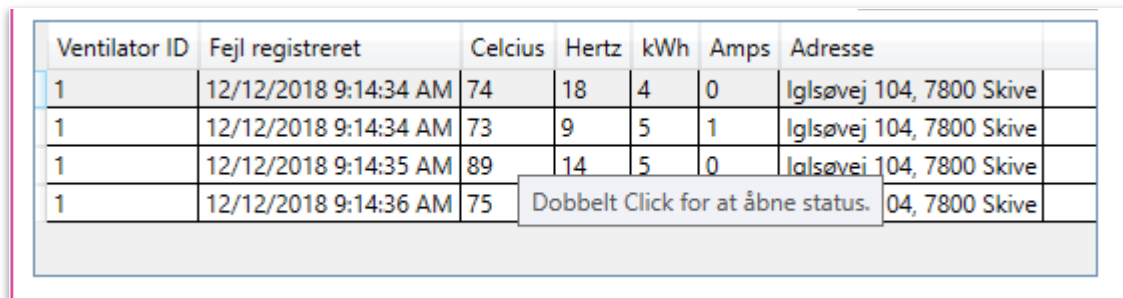
#9	Opdatere liste efter fejlrettelse.	Når sidste fejl er rettet på en ventilator status, skal ventilator statussen forsvinde fra listen.
----	------------------------------------	--

Tabel 10. Ny Use Case.

Skulle jeg implementere Use Case #9, vil det nu være nødvendigt at kunne tjekke om alle fejl er rettet for en status før den kan fjernes.

Dette vil føre til ekstra kolonner i databasen på "ventilator_status" og en masse tilpasninger igennem applikationen. Det er en større operation at lave om på kernefunktionalitet og denne nye Use Case er ikke en jeg vil begynde at implementere så sent i projektet.

Spørgsmål 2 og 3, drejer sig om, hvordan man anvender og håndtere applikationen. Her vil det være en god ide at lave en instruktionsmanual/brugervejledning til applikationen eller lave undervisning til instruering af personale. Jeg har dog på baggrund af denne test lavet Tool Tips Figur 28 for at guide, brugeren af applikationen, i den rigtige retning.



Ventilator ID	Fejl registreret	Celcius	Hertz	kWh	Amps	Adresse
1	12/12/2018 9:14:34 AM	74	18	4	0	Iglsøvej 104, 7800 Skive
1	12/12/2018 9:14:34 AM	73	9	5	1	Iglsøvej 104, 7800 Skive
1	12/12/2018 9:14:35 AM	89	14	5	0	Iglsøvej 104, 7800 Skive
1	12/12/2018 9:14:36 AM	75	Dobbelt Click for at åbne status.			04, 7800 Skive

Figur 28. ToolTip eksempel.

Perspektivering

Under projektforløbet er jeg løbende støt på flere funktionaliteter, her blandt Use Case #9 fra alfatesten i transition, som jeg kunne se at applikationen kunne drage nytte af, på den ene eller anden måde, men på grund af projektets størrelse og den afsatte tid er disse funktionaliteter blevet arkiveret, jeg vil her beskrive de funktionaliteter jeg har støt på.

- Ventilatorer modeller
 - Fordi der selvfølgelig er mange ventilatorer på markedet, kunne funktionalitet der beskriver ventilator ned i model type, være en fordel.
- Ventilatorer Niveauer
 - Med niveauer mener jeg en form for ventilator management platform. Her kunne man med niveauer, inddele ventilatorer således at en ventilator kan høre til en gruppe af ventilatorer i et afsnit af en parkeringskælder, som igen tilhører en bestemt parkeringskælder osv. Dette ville bevirke over for en service tekniker at han hurtigt kan finde frem til en lokation for en ventilator.
- Ventilator komponenter
 - Funktionalitet hvor man direkte kan angive om en komponent eller flere komponenter er skyld i at en type fejl opstår.
- Reservedelslager.
 - Funktionalitet som at have reservedelslager koblet på applikationen, så en ventilator model får vist de reservedele der tilhører netop den model som ventilatoren har. på den måde er det nemt at vælge de rette reservedele til netop den ventilator der skal have skiftet en reservedel.
- Business Intelligence.
 - Ved at lave et business Intelligence projekt på de opsamlede data, kan man lære noget nyt omkring ventilatorerne, her kan man med stor sandsynlighed se hvilken status en ventilator har haft op til en fejl, på den måde kan man så gribe ind før ventilatoren får et nedbrud og

- Optimering af materialevalg baseret på historik og producent.
 - Optimering af materialevalg, går ud på at vælge den rette reservedel.
- Service dokumentation(billeder).
 - Funktionalitet, hvor dokumentationen foregår via billeder, disse billeder af fejl og rettelser kan gemmes på databasen, så billederne kan bruges til at lave billedgenkendelse af både fejl og rettelser for at verificere at rettelser bliver udført ens eller at en type fejl ofte opstår.
- Sikkerhed ved brug af applikationen.
 - Funktionalitet der sikre at en medarbejder har et login.
 - Funktionalitet der sikre at Novencos administrator funktion til at sætte grænseværdier også er sikret med et login, da det er en kritisk funktionalitet.

Konklusion

Jeg har i denne eksamensopgave givet et bud på og vurderet hvorledes opsamlet data fra ventilatorer kan anvendes og efterfølgende arbejdes med, for at bidrage til reducere omkostninger og forlængelsen af levetiden, for Novencos produkter.

Hvordan kan jeg udvikle en applikation, som kan notificere servicetekniker om ventilations data, der overstiger Novencos grænseværdier?

Jeg har i dette projekt udviklet en prototype applikation, applikationen hoved funktionaliteter er rettet mod en service tekniker. Applikationen kan notificere serviceteknikeren, omkring data (ventilator status) som overstiger en grænseværdi sat af Novenco. Disse grænseværdier kan Novenco selv bestemme og justere efter behag.

Hvordan kan jeg opsamle serviceteknikerens fejlrettelser fra en applikation?

Serviceteknikeren kan se ventilator statusser i applikationen, han kan herefter navigere ind på en bestemt type fejl i applikationen og afrapportere sine observationer omkring fejlen og hvilke tiltag han har gjort for at rette fejlen, dertil kommer at han kan tilføje reservedel til denne fejlrettelse, som alt sammen bliver persisteret til databasen. Serviceteknikeren kan også indrapportere fejlrettelser på en ventilator uafhængig af om ventilatoren har genereret en ventilator status.

Hvordan kan back end løsningen afrapportere generelle fejlrettelser til virksomheden?

Back end løsningen er databasen. Og gennem den er det mulig at afrapportere fejlrettelse til virksomheden. Den indeholder alle data på en ventilator fra ventilatoren bliver oprettet i systemet og den efterfølgende drift af ventilatoren. Her er der lagt vægt på at registrering af fejlrettelser, da det alt andet lige har indvirkning på ventilatorens levetid og omkostninger.

Denne løsning gør det muligt at kunne spare, på hel eller halvårligt eftersyn, ved kun at reagere på fejl, når de opstår. Således kan man spare på driftsomkostninger til eftersyn, fordi man ikke skal ud og tjekke en ventilator med normal status.

Med denne løsning er det også muligt at forlænge levetiden for en ventilator, ved at reagere på fejl statusser kan man undgå at en ventilator kører sig selv i stykker, før dens forventede levetid er slut.

Ved anvendelsen af denne løsning vil man begynde at skabe en mængde data, dette data er grundlaget for at kunne analysere sig frem til yderligere reducere af omkostninger og forlænget levetid. Ved at analysere på data, der bliver genereret af denne applikation, vil sådan en analyse som f.eks. gennem et Business Intelligence projekt, kunne give indsigt i mange aspekter af en ventilators omkostninger og levetid. Man kan derfor begynde at tænke på begreber som forudsigelig vedligeholdelse (Predictive Maintenance), Optimering af ventilator komponenter osv.

Bilag

Dagbog

Dette er min dagbog jeg har ført den på Github og den er skrevet i Markdown, Markdown er et letvægts mark-up-sprog, med almindelig tekstformatering syntaks.

Link til Github: <https://github.com/bent-mortensen/Afsluttende-eksamen>

Dagbog

[Back](https://github.com/bent-mortensen/Afsluttende-projekt#top)

Denne dagbog er lavet for min egen skyld så jeg kan huske hvad jeg har lavet i løbet af projektforløbet.

Projekt forløbet starter mandag i uge 43 og slutter fredag i uge 51.

jeg kan samtidig holde den op på den tidsplan jeg har udformet i inception fasen.

Indhold

* [Uge 43](#uge43)

* [Uge 44](#uge44)

* [Uge 45](#uge45)

* [Uge 46](#uge46)

* [Uge 47](#uge47)

* [Uge 48](#uge48)

* [Uge 49](#uge49)

* [Uge 50](#uge50)

* [Uge 51](#uge51)

Uge 43

Uge 1.

Mandag den 22-10-2018

Mandag havde vi kick off på eksamensprojektet. Ove har lavet en slide i powerpoint til rapport inspiration. Denne skal lige indhentes.

Jeg har præsenteret Ove for min problemstilling/problemformulering og han havde kun nogle mindre ændringer til denne.

Forklar hvem kaastrup|andersen er? samt

En mere beskrivende problemformulering, over hvad det egentlig er jeg vil lave, da den allerede formulerede problemformulering

var for abstartet.

Eksamensdag

Jeg fik bestået eksamen med et flot 10 tal. Klaus ville dog gerne at jeg blev mere klar på hvilke udviklingsmetoder der passer

på forskellige projekter/senarier. Jeg fik indtryk af at dette var et generalt emne på dagen.

Tirsdag den 23-10-2018

Jeg startede dagen med at omskrive min problemstilling/problemformulering.

Jeg har lagt en grov tidsplan over forløbet. Jeg finder frem til forskellige emner jeg skal berører i projektet.

Jeg har besluttet at bruge unified process til udviklingsmetode samt at anvende usecases til mit projekt.

Jeg har lavet en template for min dagbog.

Jeg har lavet en tidsplan for forløbet i google spreadsheet

[Tidsplan](<https://docs.google.com/spreadsheets/d/e/2PACX-1vTmRW2VOF8tnhqKys-RJZ20enCPFaikzYS86ycknM110pRSiKs54lquYqlihYrrJZRRyb9z2On83Is/pubhtml>)

Ove godkendte min problemstilling/problemformulering kl 12.01

Onsdag den 24-10-2018

Jeg er nu rykket ud til kaastrup|andersen for at lave og skive mit afsluttende eksamens projekt/rapport.

Der er imellem tiden blevet rykket rundt på arbejdspladserne så nu sidder vi neden under hvor der er meget mere fred og ro til at arbejde.

Jeg har i dag arbejdet med Min domæne model As-is og To-be. Jeg har også fået skrevet lidt rapport afsnit til disse to modeller.

Jeg vil gerne have mere fra hånden end jeg har fået lavet i dag.

Novenco spørgsmål.

* Hvordan ser en typisk Service aftale ud for Novenco?

* Laver de en periodisk service eftersyn/kontrol af ventilatorer? halvårligt helårligt efter behov?

telefonnummer til den danske afdeling. +45 70 77 88 99

Torsdag den 25-10-2018

Julefrokost email er blevet sendt til min main mail.

Musik til dagen var [soundhound](<https://soundcloud.com/trancepsyberia/trance-psyberia-live-avalon-hollywood-ca-09082018>).

Jeg har skrevet virksomhedsanalyse på Novenco.

Jeg har lavet rapport afsnit til domæne modeller.

Dagsorden for imorgen

- * Jeg skal have lavet kardinalitetsforhold på de to domæne modeller.

- * Projektstyring afsnit til rapporten

- * Metode valg

Fredag den 26-10-2018

Jeg er gået igang med at skrive omkring metode valg.

Jeg har skrevet et afsnit om projektstyring til rapporten, heri ligger nogle af min valg for projektet og hvordan jeg har tænkt mig at løse dette projekt.

Overvejelser - Fredag er en travl dag i kaastrup|andersen, så jeg skal ikke have de store planer for dagen hvis der skal arbejdes seriøst eller også skal jeg finde en anden lokation til at lave mit arbejde. Der er simpelthen for meget larm og uro.

På mandag skal jeg have afsluttet og færdiggjort mange af de rapport afsnit jeg er påbegyndt.

Domænemodel skal laves færdig-!-!-!-!!!!!!

[To the top](#top)

Uge 44

Uge 2.

Mandag den 29-10-2018

Jeg vil starte med at lave min domænemodel om.

Jeg har lavet rapport afsnit i google docs, men disse skal jeg have overført til MS word.

Jeg har lavet domæne modellerne om så de nu indeholder kardinalitetsforhold. og det er blevet meget mere overskueligt og lettere at forstå.

Jeg har startet mit word dokument op og lært at lave billed tekst.

Jeg har startet på Brief Use case afsnittet.

Jeg skal have lavet en prioriteret liste af mine brief use cases, til når jeg skal lave applikationen.

Jeg har brugt Min vejleder i dag til at få svar på om der skal ligge en virksomhedsanalyse på k|a. det skal der ikke.

Jeg er ikke begyndt på noget af det som står på denne uge tidsplan, hvad angår database mapping og entity relations diagram.

I morgen vil jeg -

Starte op på ERD for databasen.

Starte med at skrive om ERD og databasen.

Tirsdag den 30-10-2018

Jeg går igang med at lave ERD på draw.io

Onsdag den 31-10-2018

Det går langsomt med at lave ERD. men tror jeg er færdig... det kan godt være at der skal ændres i den senere. men indtil videre vil jeg mene den er ok og kører klar.

Jeg har startet med at lave Mapping og har en næsten færdig model.

Jeg vil starte dagen imorgen torsdag med at skrive omkring ERD og Mapping.

Begge diagrammer skal med i rapporten og laves pæne at se på.

Torsdag den 1-11-2018

jeg har startet med at skrive omkring ERD til rapporten

jeg har tilpasset ER diagrammet så det passer på en A4 side og er mere spiseligt, samt ændret i navnene så de skaber en bedre forståelse af diagrammet.

Mapping skemaet er også blevet rettet til med navne, så det passer med ER diagrammet. Mapping skemaer er altid pæne at se på, så der er ikke gjort så meget.

Jeg ved ikke om jeg skal møde ind imorgen, pga af støj og mange mennesker.

Fredag den 2-11-2018

Jeg valgte at møde på k|a i dag. jeg har arbejdet med database modeller hele ugen og det har taget længere tid end jeg har regnet med.

Jeg har dog fra start af, i min tidsplan, sagt at noget af arbejdet måske kommer til at gå ind i næste uge og at der vil komme en glidende overgang mellem de to iterationer.

[To the top](#top)

Uge 45

Mandag den 5-11-2018

Jeg startede dagen med at lave nogle smårettelser til ERD og Mapping. Jeg kan se at jeg sagtens kunne gå meget dybere ned i detaljer med tingene. men det vil mere være min egen forfængelighed at tingene skal

være i orden, og det vil ikke nødvendigvis give mere værdi til projektet at jeg sidder og laver rettelser konstant.

Jeg har også fået rettet min rapport til og sat billeder ind.

For straks at få afsluttet forige iteration med Inception vil jeg gå igang med at lave DDL for min database.

Tirsdag den 6-11-2018

Jeg er gået i gang med at lave Mock data til databasen.

Jeg skal have skrevet noget til rapporten omkring

- * databasen

- * afgrænsninger/fravalg. som f.eks.

- * at have forskellige modeller af en ventilator.

- * carpark grupper

jeg vil i morgen kigge alle punkter igennem og se om der skal skrives noget mere på det jeg har arbejdet med indtil nu.

der skal laves lidt mere struktur på rapporten.

Iterationerne skal beskrives bedre og have deres egen overskrift i rapporten.

Jeg vil lave iterationsafsnit som enkelt stående afsnit.

Onsdag den 7-11-2018

Jeg har kigget min rapport igennem og fået mere struktur på den.

Jeg har sendt mit arbejde til Ove, med hensigt til at få noget vejledning på fredag.

Jeg er efter middag gået igang med at lave templates til mockup af mit program.

Jeg ved ikke om jeg skal fully dress mine use cases nu eller om jeg skal gå igang med at designe applikationen nu.

Jeg tror at en kombination af dem begge vil være at foretrække.

Torsdag den 8-11-2018

Onsdag var en træls dag... jeg kunne ikke lige sætte en finger på hvad der var i vejen, men jeg fik ikke lavet mockups af programmet.

Jeg gik tidligt hjem onsdag...

Jeg startede dagen med at lave fully dressed use cases.

Jeg gik tilbage og lavede en prioritering af mine brief use cases. Dette var nødvendigt, for at få fat i de mest kritiske dele af programmet først.

Jeg har lavet Mockup af de tre prioriterede Brief use cases som også er blevet fully dressed, mere eller mindre. de er ikke så beskrivende som jeg tidligere har lavet, men de er også barberet helt ind til benet.

Fredag den 9-11-2018

Vejledning Fredag... bliver spændende at få noget feedback!

Med udgangspunkt i Ove feedback har jeg omstruktureret min rapport til det bedre.

Jeg skal have skrevet et lille afsnit omkring hvad der er sket i overgangen mellem insection og elaboration.

[To the top](#top)

Uge 46

Mandag den 12-11-2018

Jeg vil gå igang med at få en app op at kører og begynde at designe appen ud fra mockups

Jeg har fået hul igennem til databasen.

Tirsdag den 13-11-2018

Jeg har arbejdet på skolen i skive i dag. Fik taget portrætbillede i dag.

Jeg har hovedesageligt arbejdet med rapporten tirsdag og fået skrevet en del på denne.

Onsdag den 14-11-2018

Jeg vil lave en SD for Use Case #1 i Draw.io.

Jeg har lavet SD use case #1..

Og jeg er begyndt at bygge applikationen op med classer objecter og database struktur osv.

Torsdag den 15-11-2018

Jeg har været med til retrospektive i dag i k|a og fremlagt hvad jeg har lavet på min rapport ind til videre.

Jeg vil fortsætte med at kode på programmet i dag.

og lave et rapport afsnit til SD.

Design Studio MethodPitch <https://vimeo.com/37861987>

Fredag den 16-11-2018

Jeg har i dag startet med at hører et oplæg fra danskebank og deres mobilepay og deres tilgang til agile i dere organisation.

Jeg skal have kodet noget mere i dag.

Jeg måtte ændre min database i dag for de tabeller i ventilatorstatus og sap havde ens navngivning og mit program kunne derfor ikke adskille de to navne i et table der var blevet genereret ud fra an sql joins statement.

[To the top](#top)

Uge 47

Mandag den 19-11-2018

Jeg skal se denne video omkring Design Studio Method Pitch <https://vimeo.com/37861987> og så skal jeg indkalde til møde onsdag eller torsdag. husk at skrive klare noter.

--Noter her under--

Create Pitch Critique

koncept design brainstorm

Team af forskellig størrelse(personer) business, designer, udvikler, marketing

Værktøj - ingen computer.

A4 papir

Tudser

tidstagning af processen

kører denne process igennem et par gange for den samme Use Case

6, 8, 5

6-8 tegninger, 5 min. til at tegne designs 3 minutter til at pitche sine designs, 2 min til critique.

Pitch process

begynd hvilket senarie/Use Case jeg har valgt.

Hvad er mine goals

hvordan løser mit desing disse goals.

critique

Ingen taler under pitchen.

tænker på spørgsmål.

2 min at give kritik - ingen dårlig kritik. fortæl 2-3 måder hvor dette design løser problemet.

..... fortæl 1-2 måder at der kan forbedre designet.

Der må gerne stjæles, men hvis der stjæles bør man forbedre det.

1 times session.

#3

Indberette fejlrettelse på en ventilator.

Brugeren skal kunne indberette, hvad årsagen til fejlen skyldes og hvilke tiltag der er blevet taget for at afhjælpe fejlen.

#6

Oprette yderlig fejlrettelse på en ventilator med flere fejl.

Brugeren skal have mulighed for at kunne oprette og indrapportere flere fejl på en ventilator.

#7

Oprette fejl som ikke er meldt ind af en ventilator via IoT.

Brugeren har behov for at kunne indberette ekstra fejl, som er opdaget ved synlig inspektion af en ventilator.

Jeg har Ryddet op i koden og fået lavet mere struktur her databasen ved at opdele den i CRUD. Create indeholde alle INSERT statements og Update indeholder alle UPDATE statements og så videre.

Jeg har fået kodet en masse i dag men også stødt på problemer jeg skal prøve at løse i morgen.

Tirsdag den 20-11-2018

Jeg har lavet en masse på applikationen i dag.

alle mockups er blevet lavet med guid interface og essentielle dele af code behind er blevet programmeret op.

Jeg har ikke skrevet så meget til rapporten da det meste er gået med programmering.

Onsdag den 21-11-2018

Jeg fortsætter med at skrive kode..

Jeg har nu overskredet min tidsplan med en uge og 2 dage. Det har været træls og en tilbage vendende stress faktor...

Jeg har planlagt at lave Design studio method med k|a på 3 Use cases. torsdag. men det kommer til at gå ud over min produktivitet og jeg vil derfor arbejde på nogle af de simple use cases som jeg har prioriteret sidst i forløbet. På den måde kan jeg måske indhente noget af den overskredne tidsplan.

Jeg har i dag fået lavet iteration #3 på rekord tid. Jeg har været ned i en bakkedal men er kommet godt igen idag.

Torsdag den 22-11-2018

Jeg har afholdt Design studio med k|a i dag. hvor vi fik genereret omkring 30 forskellige designs til at løse en Use Case.

der er genereret nok designs til at jeg hurtigt kan lave designs i WPF

Fredag den 23-11-2018

Fredag har jeg afsluttet Iteration #2 Jeg har analyseret lidt på resultatet af Design studio Method og lagt en plan for hvad jeg skal arbejde med Mandag i næste uge

Jeg starter med at lave rettelser til Databasen og få dette dokumenteret i rapporten.

[To the top](#top)

Uge 48

Mandag den 26-11-2018

Jeg er gået igang med at lave rettelser til på ERD Mapping og mit database script samt lave lidt mock data til databasen.

Jeg skal have lavet billeder til rapporten med ændringer. og beskrevet disse ændringer. database relateret.

Jeg har skrevet En masse til rapporten denne mandag og jeg skal i gang med at skrive noget kode imorgen eftermiddag tirsdag morgen vil gå med at læse rapport og få billeder til DSM afsluttet med bilag og rapport afsnit.

Tirsdag den 27-11-2018

Dagen er gået med at skrive kode til applikationen.

Onsdag den 28-11-2018

Dagen er gået med at skrive kode til applikationen.

Torsdag den 29-11-2018

Jeg er fortsat med at skrive kode til applikationen. og så har jeg udfyldt dagbogen for de sidste tre dage.

Fredag den 30-11-2018

Jeg er gået fast i en metode i applikationen og jeg kan umiddelbart ikke lige få hul på den

Julefrokost med firmaet kl 18:30 horsens hotel.

[To the top](#top)

Uge 49

Mandag den 03-12-2018

I dag prøver jeg igen at få hul på metoden... Jeg har selvfølgelig glemt det stykke papir med sudo kode på så det må vente til imorgen.

Jeg vil kigge på rapporten istedet. og måske lave en anden use case

Tirsdag den 04-12-2018

ændre afsnit med manglende motivation er nok en god ting i forhold til at tidsplanen er skredet

Bedre beskrivelse af DSM

fordele og ulemper ved DSM

funktionalitet til uc #3 bedre beskrevet

ny iteration: test generelt

bruger test, uni test

transition plan aflevering af produkt beskriv fasen

Onsdag den 05-12-2018

rapport skrivning massere af rapport skrivning

Fik lavet lidt om i applikationen efter jeg opdagede at noget af min kode ikke var object orienteret.

Torsdag den 06-12-2018

Jeg har lavet unittest af nogle metoder og fået skrevet noget rapport

blandt andet afsnit om testning f.eks. integrationstest.

Fredag den 07-12-2018

Jeg har skrevet på opsummeringen.

[To the top](#top)

Uge 50

Mandag den 10-12-2018

rapport Refaktorering konstruktion afsnit

Tirsdag den 11-12-2018

rapport kigget lidt på kode lavet alfatest af programmet

Onsdag den 12-12-2018

rapport lavet forslag til ændringer til applikationen.

opdateret min tidsplan

Torsdag den 13-12-2018

rapport og vejledning.

påfører normalsider på forsiden. til sidst.

opsummering -hedder unified process tjek.

DSM værktøj til metode gør det klart at det er en styret brain storm process til at generere designs. tjek.

genbesøg refaktoring navngivning... ændres... opdatering tjek.

ny erd og mapping skal refereres bestem i tekst at der forligger fuld figur i bilag. tjek.

tilføje use cases hvor jeg mangler arbejde på i elaboration iterationer.

construction - liste use cases som mangler arbejde, gør det klar at der ikke bliver lavet nye use cases.

construction 1 iteration eller fase pointere at meget kode arbejde er lavet i elaboration

construction vælge en anden løsning

spin på GRASP - SOLID

hvad er en alfatest brugerperspektiv

perspektivering

prototype

version 2 anden løsning høj kobling design patterns lav kobling

konklusion besvare spørgsmål fra problemstillingen vigtigt.

Fredag den 14-12-2018

rapport

[To the top](#top)

Uge 51

Mandag den 17-12-2018

rapport

Tirsdag den 18-12-2018

rapport

Onsdag den 19-12-2018

rapport

Torsdag den 20-12-2018

rapport

Fredag den 21-12-2018

rapport

[To the top](#top)

Tidsplan

Dette har været min tidsplan for projektet den er delt op i Figur 29 og Figur 30, så man bedre kan læse den. Den er lavet i Google Sheets for at have nem og hurtig adgang til den online.

1. del.

Projekt tidsplan ☆

File Edit View Insert Format Data Tools Add-ons Help Last edit was yesterday at 3:05 PM

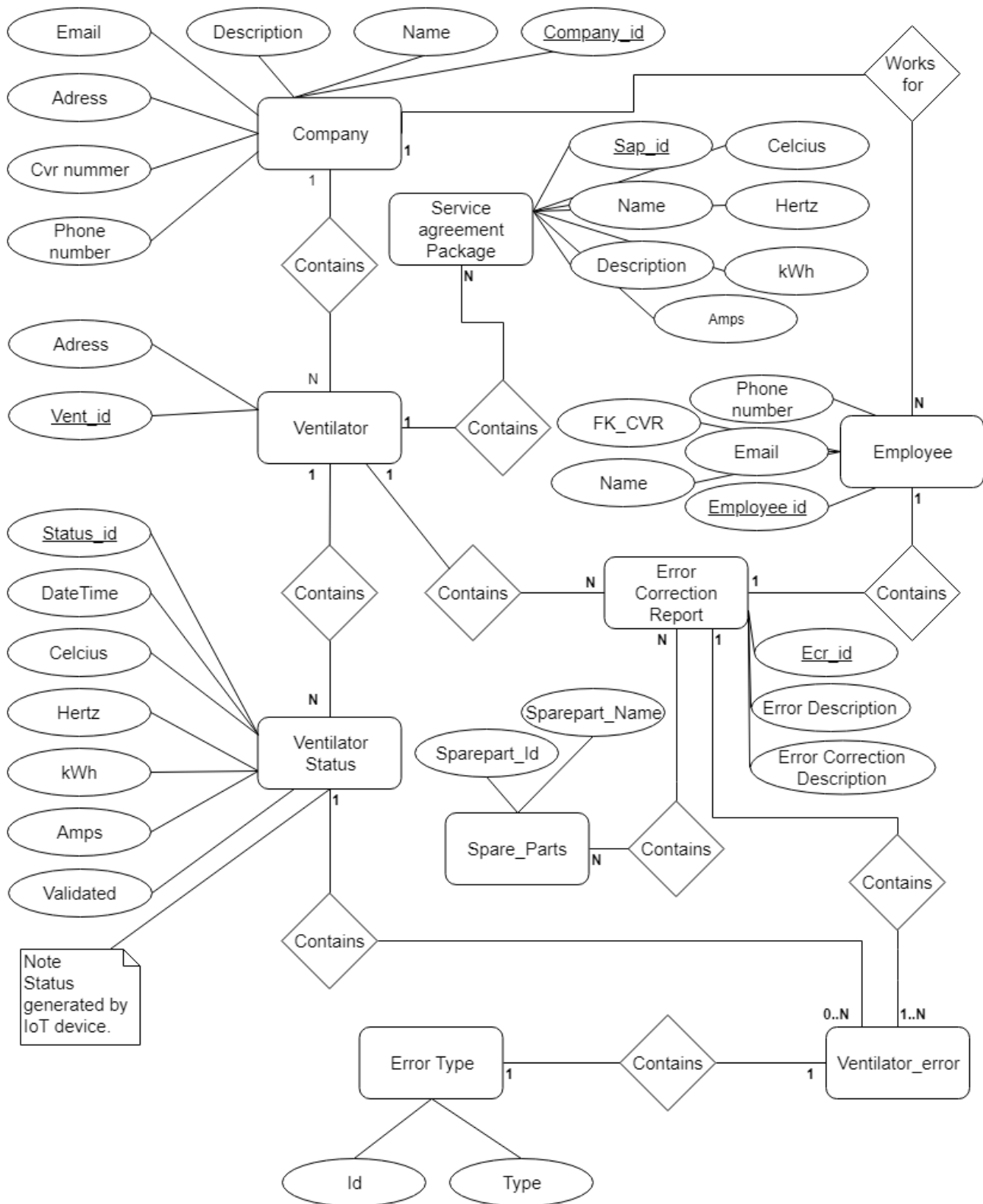
100% \$ % .0 .00 123 Arial 24 B I S A

fx Tidsplan

	A	B	C	D	E
1	Tidsplan				
2	week	43	44	45	46
3		Inception	Inception	elaboration	elaboration
4		Tidsplan - vejledende	Brief UC	Skriv kode	Skriv kode
5		Github	Entity Relationship Diagram	Fully dressed usecase	Fully dressed usecase
6		Brief UC	Mapping	Mockups	Mockups
7		Domænemodel	Data definition language	Sekvens diagram	
8		Research på valgte udviklingsmetode	Research på valgte udviklingsmetode		
9					
10					
11	Rapport	Afsnit	Afsnit	Afsnit	Afsnit
12		Dagbog	Dagbog	Dagbog	Dagbog
13		Problemstilling / Problemformulering.	Mapping	rapport	rapport
14		Virksomhedsanalyse	DDL	iteration	iteration
15		* kaastrupjandersen? Ikke nødvendig.	ERD	iteration resultat	iteration resultat
16		* Novenco	Lav forside Mandag og slut glem den indtil den 21. D.	iteration revaluering	iteration revaluering
17		Versionsstyring - Github			
18		Udviklingsmetode - Metodevalg			
19		* Unified process			
20		Domænemodel			
21		Brief usecase list			
22					
23					
24		Status	Der kommer muligvis til at ske en glidende overgang til elaboration fasen. da jeg mangler to dage på Grund af praktik eksamen og endelig godkendelse af problemstilling/problemformulering. No Worries.		
25		Færdig			
26		Igang			
27		Ikke påbegyndt			
28					

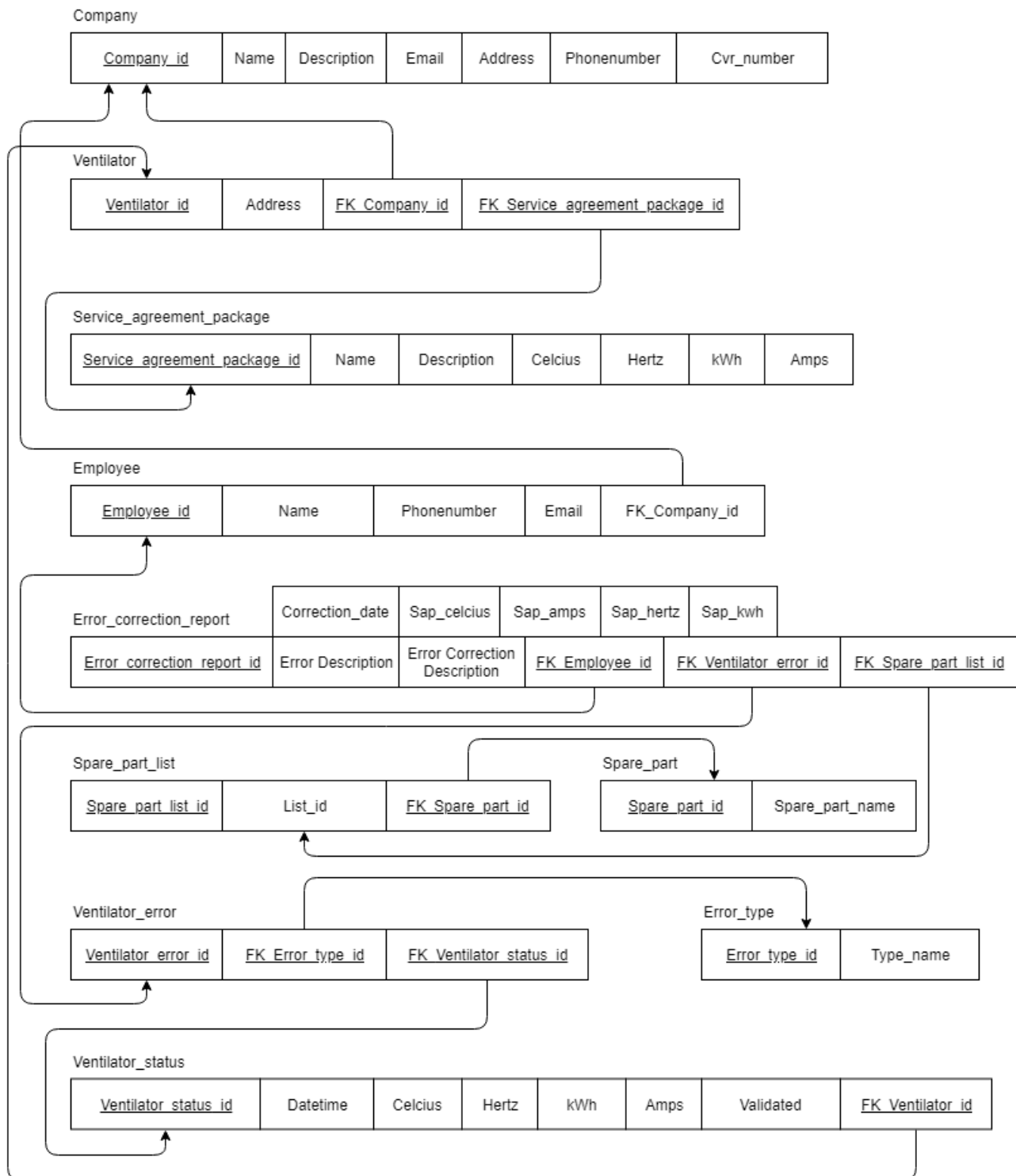
Figur 29. Tidsplan 1. del.

Entity Relation Diagram opdatering



Figur 31. Entity Relation Diagram.

Mapping skema opdatering



Figur 32. Mapping Skema.